1-1-2001

Feature based recognition of incomplete CAD models for providing design assistance

Aditya Abhinandan Ajmera
Iowa State University

Feature based recognition of incomplete CAD models for providing design assistance

by

Aditya Abhinandan Ajmera

A thesis submitted to the graduate faculty

in partial fulfillment of the requirement for the degree of

MASTER OF SCIENCE

Major: Industrial Engineering

Major Professor: Dr. Ranga Narayanaswami

Iowa State University

Ames Iowa

2001

Graduate College

Iowa State University

This is to certify that the Master's thesis of

Aditya Abhinandan Ajmera

has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Most current CAD systems have the tools to allow users to generate models using Boolean combinations of features. While current research has explored several directions for next generation CAD systems with a wider range of applications, there has been little work in providing assistance to the user for generating the models. The present research aims at using 3D object recognition techniques to recognize incomplete CAD models and thereby determine the user's intent to facilitate model development. A system has been developed to recognize complete and incomplete models belonging to a particular category for which the system stores a construction tree that describes the sequence in which features must be added in order to generate a model of the category. The construction tree of a model is analogous to the sequence of operations that would have to be performed to manufacture the part. The input CAD model is checked against the construction tree of the object in question using certain rules to obtain a confidence level representing the similarity of the input model to the object. The rules used by the system are classified as *Shape Rules*, *Dimension Rules*, *Similarity Rules* and *Placement and Orientation Rules*. The system recognizes models belonging to the category *Gear*, with sub-categories as *Spur Gear (internal & external)*, *Rack Gear* and *Straight Bevel Gear*. Test cases are provided to display the system's competence and capability.

1 INTRODUCTION

1.1 Background

Computer aided design (CAD) systems are an essential tool for mechanical engineering designs in the manufacturing industry today. The history of CAD systems goes back several decades. Early CAD systems can be traced back to 1960's when hardware for computer graphics had been developed. The first computer-aided design applications used simple algorithms to generate 2D drawings of parts which was later extended to generate 3D wireframe drawings. While wireframe drawings with capabilities of hidden surface removal did enhance the use of CAD systems, they lacked in complete and unambiguous representation of an object due to their incapability of handling spatial addressability. The 1970's saw the development of solid modeling techniques that helped in developing factually complete, valid and unambiguous objects. Solid models stored the geometric and topologic information of a part and hence solved the problem of properly displaying the intersection of surfaces.

The present CAD systems have since then evolved to provide suitable means to the user to develop solid models using several techniques, which include boundary representation (B-Rep), constructive solid geometry (CSG), feature-based modeling, etc. The boundary representation comprises of a list of vertices that stores the 3D coordinates of all

the vertices in the model, a list of edges between the vertices and a list of faces in the model. The boundary representation is most preferred by researchers due to its simplicity and the ease with which the topological information of the model can be obtained. The Constructive solid geometry approach generates CAD models by Boolean and geometric operations of solid primitives such as cubes, cylinders, etc. The Boolean operations are unions, differences and intersections, while the geometric operations are translation, rotation and scaling of the simple shapes. This allows the user to work at a higher level of abstraction than with low-level entities like lines and arcs. Feature-based modeling is similar in context to the CSG. Feature-based design allows the designer to build a model by drawing functional features and assembling them to form the product model. CSG and feature-based design offer the possibilities for manufacturing and process planning concerns to be considered throughout the design process.

Different CAD systems supporting feature-based design have different toolsets for generation of features, though most of them classify features as *Protrusions*, *Cuts*, *Slots*, etc. A protrusion is defined as a feature that adds solid material to the model. Some of the most common type of protrusions are cubes, cylinders, cones and other simple geometries. A cut is defined as an *open* feature that removes material from the model. The cross-section of a cut does not form a complete loop. A slot is similarly defined as a *closed* feature that removes material from the model. The cross-section of a slot does form a complete loop. For the purpose of this research, cuts and slots are considered as through features unless specified otherwise. Commercial CAD packages provide sophisticated toolsets to draw, edit and delete the above-mentioned features.

Besides providing a suitable interface to add these features, CAD packages provide assistance in the drawing of these features. Assistance is usually provided by setting up of simple geometric constraints during the modeling process. These constraints are generally certain assumptions that the system uses to generate the sketch. Mathematically, these geometric constraints are translated into a set of algebraic equations and are solved simultaneously using iterative algorithms. Constraint solvers such as rule-based and graph-based have generated a reasonable amount of interest in the research community [18] and have subsequently been integrated in commercial CAD packages. A very good example of this type of assistance would be the *Intent Manager* in Pro/ENGINEER, which intelligently adds constraints to the user's drawings. The intent manager continuously makes assumptions when the user is drawing a sketch of a feature. When certain drawn entity lies within the tolerance limit of a constraint, the system automatically snaps to that constraint and thus eliminates the need for a precise sketch.

While different CAD systems offer different types of assistance to facilitate model generation, there has been little work in providing intelligent assistance to the user. Intelligent help can be made available by trying to determine the designer's final intent during the model development phase.

1.2 Research Scope and Goal

As mentioned above, there has been limited research in making the CAD systems more intelligent and user friendly. A CAD system, which can find out the final goal of the designer, can assist the designer in a variety of ways including providing help to complete the

model. Determination of the designer's intent can be done either by requesting the user for information, or by recognizing the user's incomplete drawing during the design phase.

For a CAD system to automatically generate a model it would require the user to provide large quantities of data or it would have to make several assumptions as regards to the geometry, shape and size of the model. A recognition system, though by far more complex, would be flexible in allowing the user to draw at will, while it tries to recognize the user's intent from the incomplete model. The CAD system for both the above cases, stores a database containing information about the shape of different categories of objects.

The recognition of incomplete CAD models with information specifying which features in the model are present and which are missing can be used for a number of purposes. One of them would be to create a tool wherein the system, after classifying the input incomplete CAD model, would provide assistance to the user to complete the CAD model. This could either be an automatic completion of the CAD model or an interface to provide the user with different dynamically generated options to complete the model. Other directions of downstream applications include searching through a catalogue of parts to find a part having features of required shape and dimensions. The above search could be carried out by a search engine through the Internet, thereby facilitating collaborative commerce between firms. Similar type of content-based retrieval systems have been developed by researchers for querying a large database of images like the Query By Image Content (QBIC™) developed by IBM.

The goal of this research is to develop a flexible system which can competently recognize complete and incomplete CAD models and find information about the completed

features in the model. New and creative techniques can then be developed which use this information to aid the designer in his objectives.

1.3 Literature Review

The recognition process to be discussed below is greatly influenced by present research in the fields of feature-based design, feature extraction procedures, and 3D object recognition systems. A feature in feature-based design has been defined in several ways by different researchers. A simple and precise definition put forth by Shah [14] describes a feature as a physical constituent of a model with engineering significance and which can be mapped onto a generic shape. Giacometti [7] described a feature as a semantic grouping used to describe a part. Wilson and Pratt [19] have summarized the difference of opinions in the exact definition of a feature among researchers. For the purpose of this research, features can be described as sets of information that refer to the form aspects of the part, in such a way that these sets can be used in reasoning about the design and manufacture of the part.

Feature-based design systems have several advantages over other traditional techniques for facilitating design. Dixon et al [6] have illustrated the reasoning and motivation behind feature-based design. Vaghul et al [16] have gone ahead to list out the advantages of using features to design models. One of the more significant advantages of working with features is their use in numerous downstream applications such as integration of CAD/CAM using Computer Aided Process Planning (CAPP) techniques. The present research too uses features and other concepts in feature-based design for the recognition of complete and incomplete models.

CAD systems allow designers to either create their own features, or extract features from a library by specifying the exact dimension and placement in the part. Since features can be user-made, feature extraction algorithms are used to extract features from the drawing without comparing it to any pre-defined features. Over the years, research in CAD has focused on the extraction of feature information for more advanced applications like CAPP, Design for Manufacture, Design for Assembly, content-based data retrieval, etc. The Alternating Sum of Volumes (ASV) algorithm developed by Woo and Tang [20] has been one of the most referenced works in feature extraction during the 1990's. The ASV algorithm converts a B-Rep representation of a model to a CSG representation along with extraction of features with convex decomposition. The feature extraction is carried out by representing an object by a series of convex components with alternating signs. A positive sign for a component signifies that the component is added to the part, while a negative sign signifies that the component is removed from the part.

While Woo and Tang's work was pioneering in feature extraction algorithms, it lacked in a number of ways. A major problem with ASV was that it didn't always converge and hence had limited use. Kim et al [17] have presented some solutions to the above problem. The Alternating Sum of Volumes with Partitioning (ASVP) volume decomposition system developed by Kim et al has a broader application due to its definite convergence.

In addition to the decomposition algorithms discussed above, researchers have proposed several different graph-based procedures. Graph-based algorithms convert the B-Rep of an object to a graph data-structure. The graph data-structure generally has the faces, the edges, or the vertices of the model as nodes. De Floriani [4] proposed an edge-face graph representation of a model, which is then decomposed to smaller components denoting the

features in the model. Joshi and Chang [10] have put forward the concept of Attribute Adjacency Graph (AAG) for feature extraction from a B-Rep of a solid. In an attributed adjacency graph, a face is a node and an edge is an arc. Each arc in the graph has an attribute, either 1 or 0, indicating if the faces that share the edge are concave or convex. Henderson [9] suggested a vertex-edge graph-based approach with the vertices of the solid forming the nodes in the graph and the edges connecting the vertices constituting the arcs. Ahluwalia [1] developed the dynamic feature generation method where feature extraction is carried out after addition of each feature during the design phase of the model. More recently, Qamhiyah et al [13] have proposed a technique for extracting features from a planar-faced boundary representation using a loop adjacency hyper graph.

As mentioned before, feature extraction with topological information from CAD models has importance in a number of applications. This information along with the parent-child relationships between the features also forms the necessary input to a 3D model recognition system. Most present object recognition systems recognize objects from an image with range data for use in computer vision. The image is parsed to generate a boundary representation of the model. Different recognition procedures extract either features or surfaces from the boundary representation and then carry out pattern recognition to obtain a *recognition parameter*. The recognition parameter indicates the similarity of the input image to the object in question.

3D recognition systems are classified as either model-based or function-based systems. Traditional model-based systems are shape-based and check for a shape pattern in the features of the model. They use simple parameterization of the geometric entities to obtain a general representation of the object. This general representation is essentially the

pattern that defines the object. Pattern recognition is usually done using a tree search method where nodes of the tree represent an image to model entity match. Such a tree is usually termed as the *search* or the *interpretation* tree. Binford [2] has made a detailed comparison of several different model-based system developed. Chin and co-workers [3] describe more recent developments in model-based three-dimensional object recognition for robot vision.

Some shortcomings of the model-based approach is that it is difficult for any person to supply to the recognition system an accurate description of the object which is general enough to cover all the possible variations in shape. More recent developments in object recognition have concentrated on function-based reasoning, where an object is defined by the simple entities that are needed for the object to satisfy its function. Function-based reasoning has been found to be more general than model-based approaches and is similar to object reasoning done by humans. Function-based models do not use specific geometric patterns to identify models, but rely on more abstract details such as the fundamental units needed for the object to perform its function. One of the early works in describing and supporting functional-based reasoning for computer vision systems was by Di Manzo, Trucco, Giunchiglia and Ricci [5]. Their work focused on the recognition of 3D objects in the category *chair*. Kise and co-workers [11] successfully created a system for recognizing certain hand-held tools using function-based recognition methods. More recently Stark and Bowyer [8, 15] have developed a system, GRUFF (Generic Recognition Using Form and Function), which uses a set of knowledge primitives to define the functional aspects of a particular category of objects. The GRUFF system has been extended to different categories over the years, which include the category *Furniture* and the category *Scissor*. The furniture category has sub-categories as *Chair*, *Bench*, *Bed*, *Table* and *Bookshelf*.

1.4 Chapter Summary

The following conclusions can be easily drawn from the literature described above:

1. Feature-based design is one of the most accepted methods of model development in the CAD industry.

2. An intelligent intent finder can achieve reasonable automation and reduction in design lead times during model generation phase.

3. Concepts of object recognition provide promising means to determine the user's intent by recognizing incomplete models.

The next chapter, chapter 2, introduces the concept of the model construction tree and the parent-child relationships between features. The model construction tree forms the starting point in the incomplete CAD model recognition system described in this research. Examples of the construction tree for the categories hand-held tool, pulley and gear are also presented.

2 MODEL CONSTRUCTION TREE

2.1 Introduction

For a part to belong to a particular category, it must satisfy certain geometric, functional and physical characteristics that define the category. These characteristics can be mapped onto the features that must be present in a part for it to be called a member of the category. Besides the presence of features, the relationships between these features are necessary to decidedly categorize a part. Furthermore, the category must have very general, but accurate, requirements about the features and feature-relations of its members.

One of the more important feature-relations is the parent-child relationship among the features. A parent-child relationship denotes the dependency of a child feature on its parents. A child feature is dependent on its parent features since it cannot be created without first generating all its parents [12]. This could be due to the fact that the child feature has to intersect surfaces generated by its parent features. The parent-child relations could be graphically displayed as a construction tree where each node, except for the root node, can have more than one parent. Such a construction tree in essence indicates which features must precede other features for the model to be generated. The levels in the tree thus represent levels in the design phase of the product model.

The incomplete model recognition system described below stores a construction tree for each category of objects that it recognizes. The nodes in the construction tree represent the features that make up the object and the edges represent the relations between these features. A node has attributes for the shape and size of the feature and also for the relative position and orientation of that feature. The features in the construction tree of a category form the minimal set of features that must be present in a model for it to be classified as a member of the category.

The construction tree of models belonging to a particular category need not be unique since it depends on the manner in which features are extracted by the feature-extraction algorithm. In order to overcome this problem, the feature extraction algorithm must represent the model as a Destructive Solid Geometry (DSG). A destructive solid geometry representation of a model is a special case of the CSG tree where every operation, excluding the first one is of difference type. This can be explained in the context of feature-based design as a model having an initial protrusion followed by any number of cuts and slots. Such a representation would cause the construction tree of every model to have a single protrusion at the root node and cuts or slots at any other child nodes. The destructive solid representation is analogous to the sequence of material removing operations that would have to be performed, in order to manufacture the part from raw stock.

The model construction tree forms the basis for recognizing incomplete objects. The levels in the tree are an indicator for the *completeness* of the model. The recognition process is feature-based in the sense it tries to determine the level of completeness after the addition of each feature. The model construction trees of some simple objects have been described below.

2.2 Category – Hand-Held Tool

2.2.1 Screwdriver

Among the hand-held tools, the screwdriver is universally known. There are two basic types of screwdrivers, the standard slot-head and the Phillips cross-tip. The basic functional shape requirements for the standard slot-head screwdriver are a cylindrical handle, a cylindrical shank, a broader blade and a flat tip. These functional requirements can be converted to a simple DSG tree representation. The DSG tree would have an initial cylinder as a protrusion, followed by cuts to make the shank, blade and the tool tip. Figure 1 displays the construction tree for a slot-head screwdriver. The construction tree highlights the basic features and their sequence of generation, which must be carried out to build a model of part belonging to the slot-head screwdriver category. Figure 2 displays the step-wise progress of the construction of a slot-head screwdriver model.

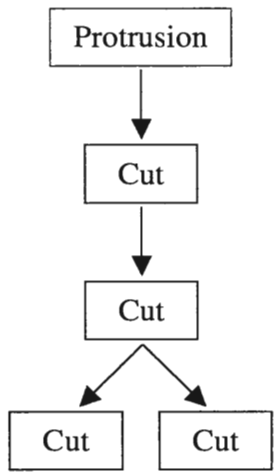
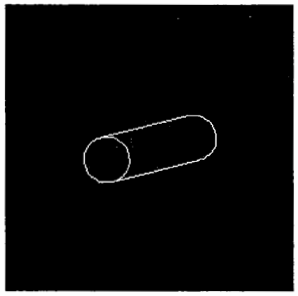
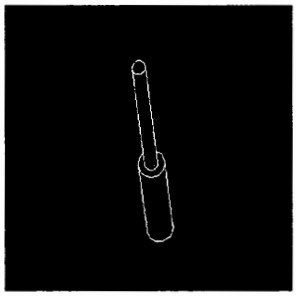


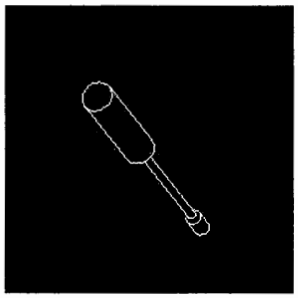
Figure 1 Construction tree of Slot-head Screwdriver



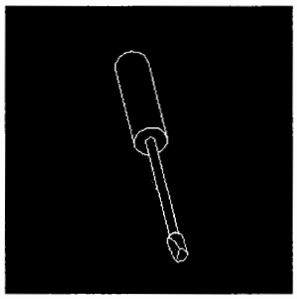
Step 1



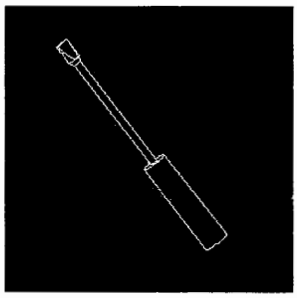
Step 2



Step 3



Step 4



Step 5

Figure 2 Construction of Slot-head Screwdriver

The functional requirements of the Phillips cross-tip screwdriver are similar to that of the slot head, with the exception that a cross-shaped tip replaces the blade and the tip. Following the cut for the shank in level 2, a conical tip is created in level 3 on which the four grooves for the cross-shaped tip are generated. Figure 3 graphically portrays each level in this screwdriver's construction tree while Figure 4 describes the construction steps that are considered necessary to generate a model of this type of a screwdriver.

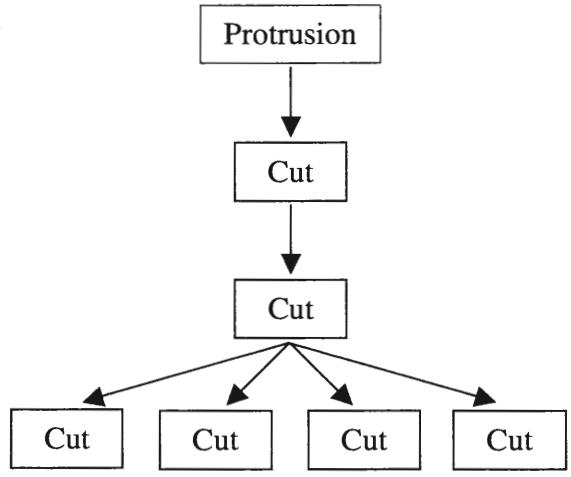
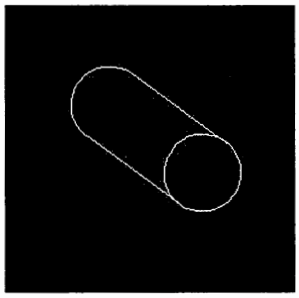
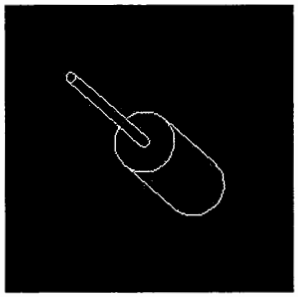


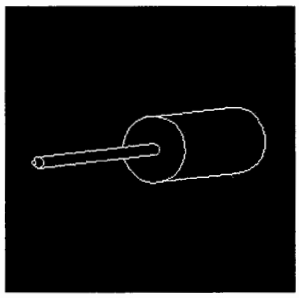
Figure 3 Construction tree of Phillips-head Screwdriver



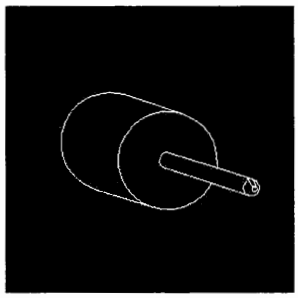
Step 1



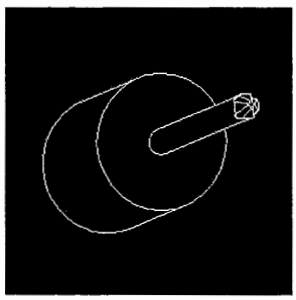
Step 2



Step 3



Step 4



Step 5

Figure 4 Construction of Phillips-head Screwdriver

2.2.2 Spanner

The spanner is a simple wrench used to tighten or loosen nuts and bolts. The functional requirements of a spanner would be a handle to hold it and a feature to grasp the nut. The DSG tree of such a tool would have a protrusion having the shape of a rectangular prism at the root node; followed by cuts to create the surfaces need to hold the nut. The construction tree diagram is clearly depicted in Figure 5, and Figure 6 displays each stage involved in creating a simple CAD model of a spanner.

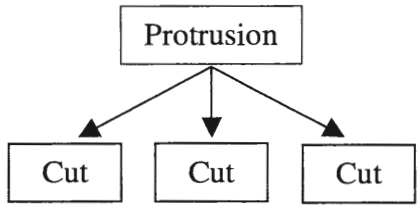
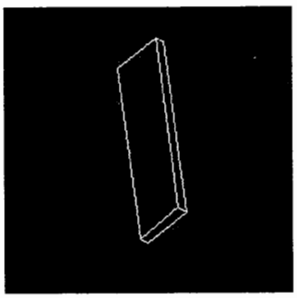
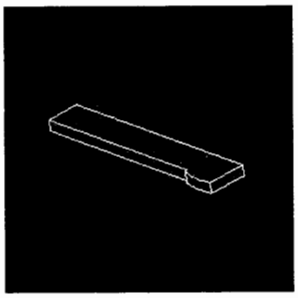


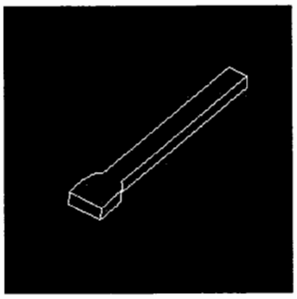
Figure 5 Construction tree of Spanner



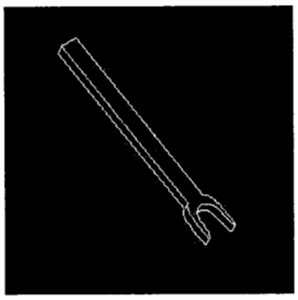
Step 1



Step 2



Step 3



Step 4

Figure 6 Construction of Spanner

2.3 Category – Pulley

A pulley is a simple machine with the shape of a grooved wheel used to raise or lower objects with the help of a rope or a cable. Functional necessities for such an object would be a cylindrical wheel with a groove on the cylindrical surface for a rope to wrap around. A through hole would also be needed to hook up the pulley with the entire mechanism. The DSG tree would comprise of a cylinder as the initial protrusion, followed by a cut for the groove and a hole for linking the pulley. Figure 7 displays the parent-child relationships while Figure 8 shows the stages involved in constructing a pulley model.

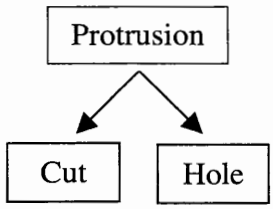
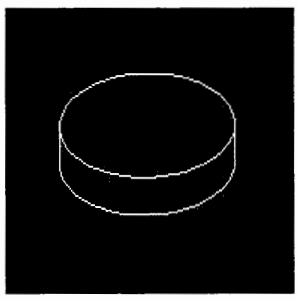
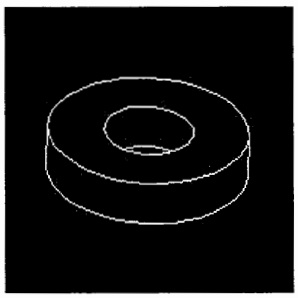


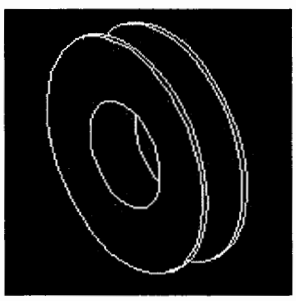
Figure 7 Construction tree of Pulley



Step 1



Step 2



Step 3

Figure 8 Construction of Pulley

2.4 Category - Gear

Gears are very commonly used mechanical devices. They are a means of changing the rate of rotation of a machinery shaft during transmission of power. They can also change the direction of the axis of rotation and can change rotary motion to linear motion and vice-versa. Gears are of several categories and can be combined in a multitude of ways. Some sub-categories of gears discussed here are the external spur gear, the internal spur gear, the rack gear and the straight bevel gear.

2.4.1 External Spur Gear

The external spur gear is of cylindrical shape with teeth on its periphery and a slot for linking it with the machine. The DSG tree, Figure 9, of the external spur gear comprises of a cylindrical protrusion at the root node followed by a through hole and a number of cuts at the nodes in the second level. Figure 10 shows the steps needed to construct a simple external spur gear model.

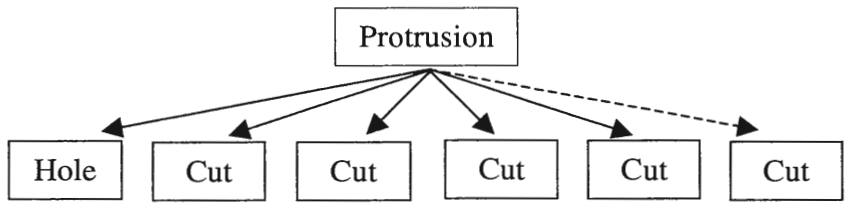
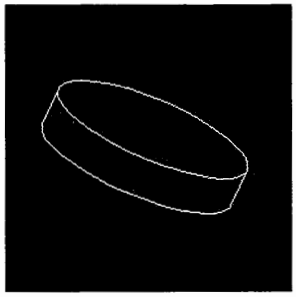
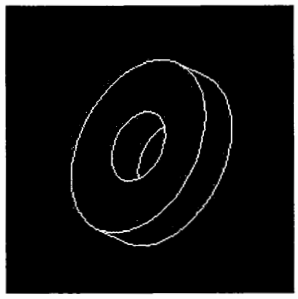


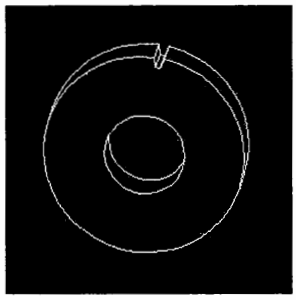
Figure 9 Construction tree of External Spur Gear



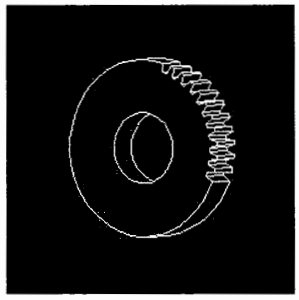
Step 1



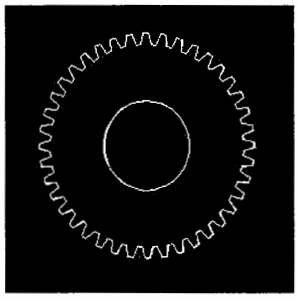
Step 2



Step 3



Step 4



Step 5

Figure 10 Construction of External Spur Gear

2.4.2 Internal Spur Gear

The internal spur gear has similar features to that of its counterpart, the external spur gear. Its shape can be generalized as having a protrusion with no specific shape requirements and a through hole where the gear teeth are located. Figure 11 helps understand the DSG tree of a model belonging to the internal gear sub-category. Figure 12 demonstrates the phases during the development of the internal gear model.

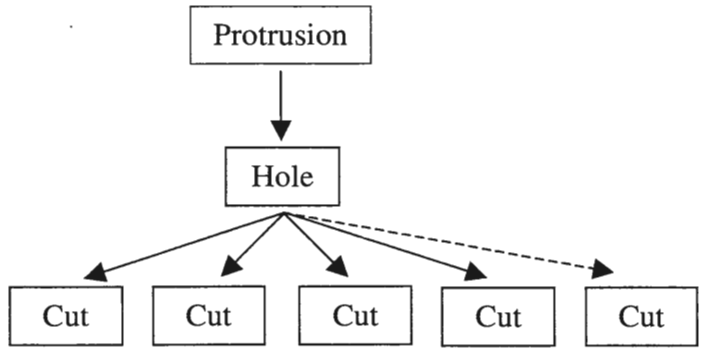
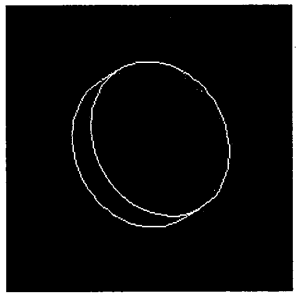
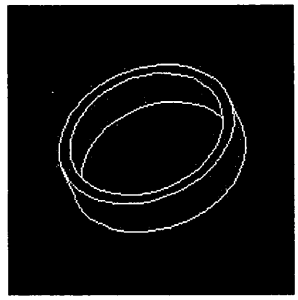


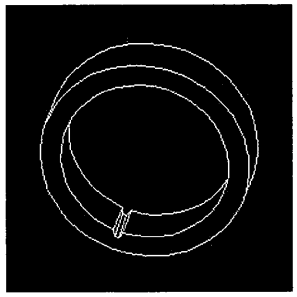
Figure 11 Construction tree of Internal Spur Gear



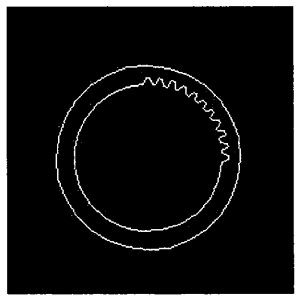
Step 1



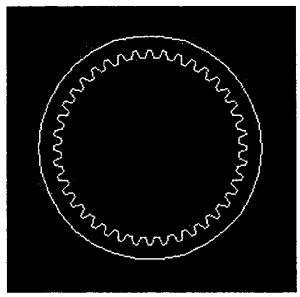
Step 2



Step 3



Step 4



Step 5

Figure 12 Construction of Internal Spur Gear

2.4.3 Rack Gear

Rack Gears are linear spur gears having a cog set in a straight line, which meshes with a pinion (external spur gear) and are used to transform linear motion into rotary motion and vice-versa. The cog set can be built by creating cuts on one of the surfaces of a rectangular prism, which makes up the root node in the DSG tree of a rack gear model. The nodes in the second level of the tree would represent the cuts (Figure 13). Figure 14 describes the details of all the development phases to make a rack gear part model.

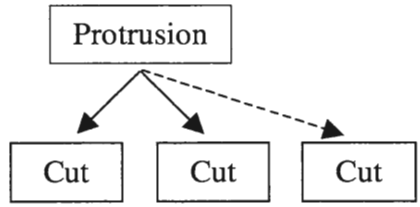
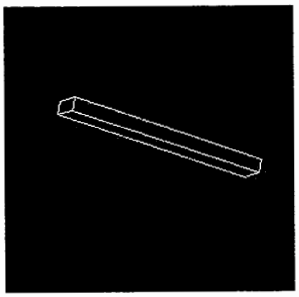


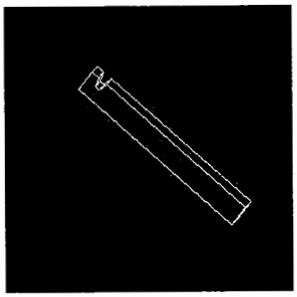
Figure 13 Construction tree of Rack Gear

2.4.4 Straight Bevel Gear

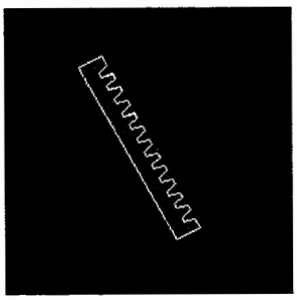
Straight Bevel Gears are useful for changing the direction of the rotation of a shaft in a machine. They are akin to the external spur gear, but with a conical shape. The parent-child relationships for a simple straight bevel gear are illustrated by the DSG in Figure 15 and Figure 16 shows model images of the corresponding steps.



Step 1



Step 2



Step 3

Figure 14 Construction of Rack Gear

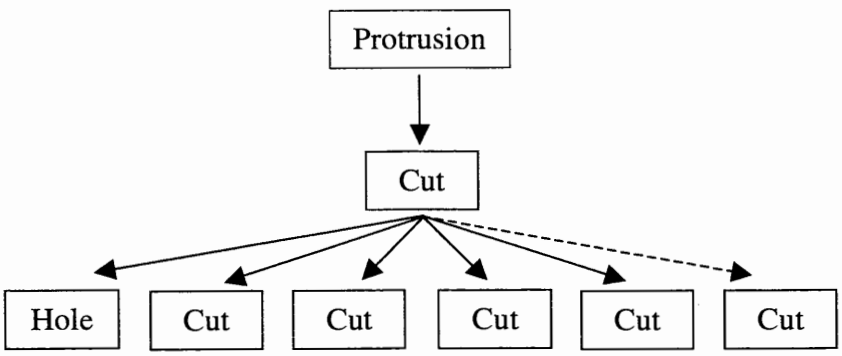
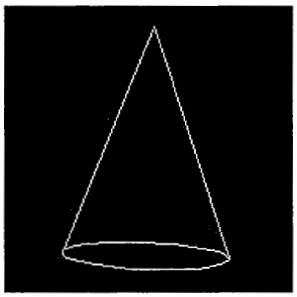
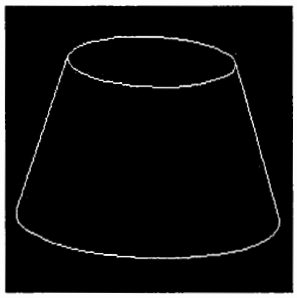


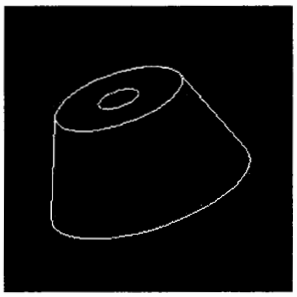
Figure 15 Construction tree of Straight Bevel Gear



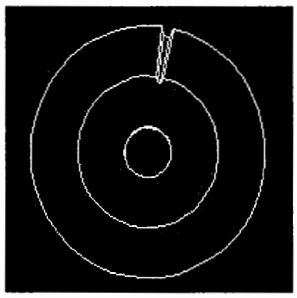
Step 1



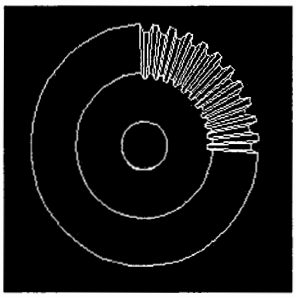
Step 2



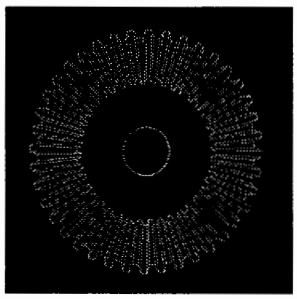
Step 3



Step 4



Step 5



Step 6

Figure 16 Construction of Straight Bevel Gear

2.5 Chapter Summary

In this chapter, the model construction tree and the manner in which it is generated for different categories was discussed. Construction tree examples were provided for the categories, hand-held tool, pulley and gear along with their subcategories. The subsequent chapter describes the recognition rules and their classification together with certain relevant examples.

3 RECOGNITION RULES

3.1 Introduction

This chapter describes in detail the general aspects needed to develop an intelligent system that can interpret a user's complete as well as incomplete CAD drawing and classify it among a set of different categories for each of which it stores a model construction tree. Such a system would require concepts in object classification and recognition to identify an input CAD model.

As mentioned earlier, for a system to claim that an input model is a member of a particular category, it must at first have information about the shape and relationships among the features that explicitly define the category. With parent-child relationships among features having been discussed in the previous chapter, this chapter talks about the shape and other feature relationship requirements that could exist for a part to belong to a specific category. These requirements could be perceived as certain hints or rules that must be reasonably satisfied for the grouping of the input CAD model.

In order to generalize the procedure of construction of these rules, they are classified into four broad groups: 1) Shape Rules, 2) Dimension Rules, 3) Similarity Rules and 4) Placement and Orientation Rules.

3.2 Shape Rules

Shape rules, as the name implies, are used to check the geometry of the feature. The geometry is checked by determining if the surfaces that make up the feature are of the right profile and in the proper position. More specifically, the parameters that need to be verified are the number of surfaces the feature has, the type of surfaces present, the number of edges between the surfaces, the type of edges, the angle between the surfaces, the relative dimensions of the surfaces, etc.

For instance, the protrusion at the root node of an external spur gear construction tree should be a cylinder. A cylinder is defined as a feature having four surfaces, two of which must be planar and the other two must be cylindrical. The planar surfaces should be parallel to each other and should not coincide. The axes of the cylindrical surfaces must coincide and should be perpendicular to the planar surfaces. Also the depth of the cylinder, relative to its diameter, should neither be too small to make the gear unstable, nor should it be too large to restrict its use in forming a gear (Figure 17). In the same way the protrusion in the construction tree of the rack gear must be a rectangular prism. A rectangular prism can be illustrated as a 3D rectangle having certain depth, i.e. three sets of mutually perpendicular planar surfaces with each set comprising of two distinct parallel planes (Figure 18). For the case of the straight bevel gear construction tree, the root node should contain a conical protrusion. For a feature to be labeled as a conical protrusion, it should comprise of three surfaces, two conical and one planar. The axes of the conical surfaces should coincide and should be perpendicular to the planar surface (Figure 19).

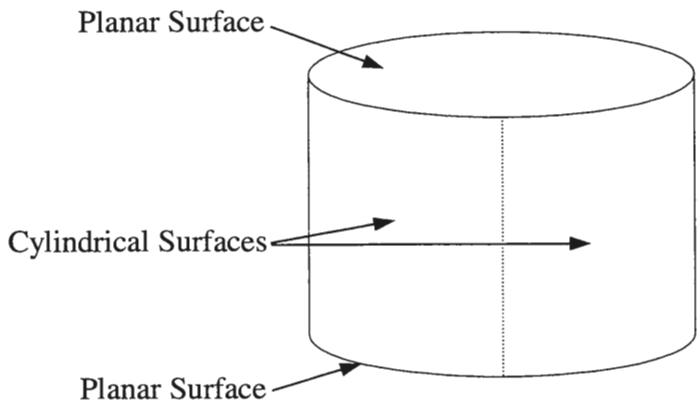


Figure 17 Cylinder – External Spur Gear Protrusion

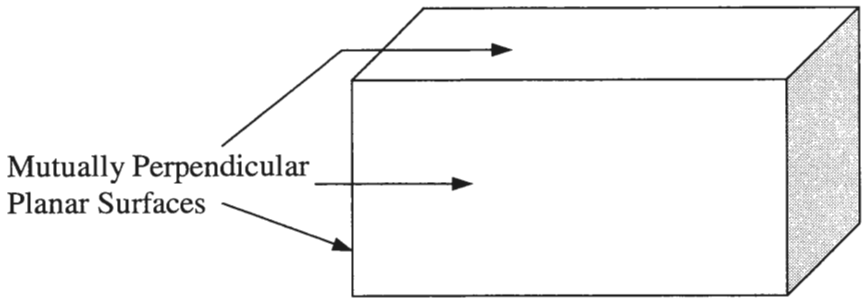


Figure 18 Rectangular Prism - Rack Gear Protrusion

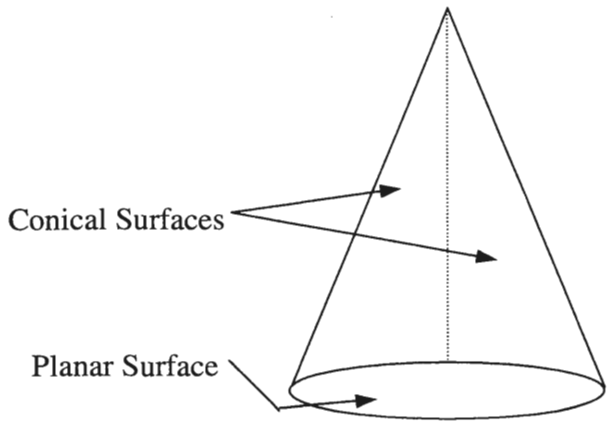


Figure 19 Cone – Straight Bevel Gear Protrusion

The cuts following the initial protrusion in the construction trees of different gears shown in chapter 2 are features needed to generate the teeth of the gear. These cuts, referred to as gear cuts, have shapes that depend on the contour of the teeth of the gear. The shape can be generalized for simplicity as a feature having three quadrilateral surfaces placed end to end. The side surfaces must be exactly similar and at equal angles with the middle surface. Figure 20 below demonstrates the cross-section of a gear cut.

The hole feature, which too is present in the construction trees of certain gears, is normally understood as one having two cylindrical surfaces. The surfaces must be similar, i.e. must have equal radii, and should have coincident axes. A cross-section of the hole feature is displayed in Figure 21.

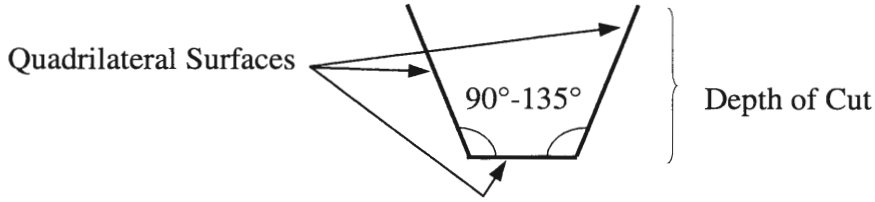


Figure 20 Gear Cut

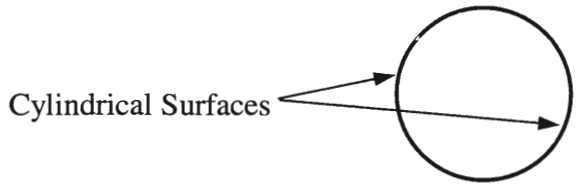


Figure 21 Hole

The shape rules help in eliminating undesirable features earlier on in the recognition process. Most shape rules are represented as functions in the system described below with the feature consisting of the input parameter and the return data types as either boolean or real number between 0 and 100. The use of certain threshold values helps conclude if the feature has passed the rule or not. The choice of the threshold values affects the average running time of the recognition system. For example, in the system developed, the angle between the side surfaces and the middle surface of the gear cut feature should lie in between 90° and 135°, with the optimum being 112.5°. The function corresponding to this rule shall return a 0 in the case the angle lies outside the above range and would return 100 if the angle is exactly 112.5°. The return value linearly decreases from 100 to 0 as the deviation of the angle increases from the optimum value.

3.3 Dimension Rules

The dimension rules follow the shape rules in the recognition process. They check for the relative dimensions of the features to ensure that they would be able to perform their respective functions. The optimum relative dimensions of the features need not be a single number and may lie within a range of values. For example, the depth of the gear cuts in an external spur gear model should not be too large for the gear to appear like a fan (Figure 35), nor should it be too small for it to be insignificant (Figure 34). Similarly the diameter of the hole in an external spur gear model, needed for linking it to a shaft, should be within reasonable ranges with respect to the diameter of the gear (Figure 32 and Figure 33).

Comparing the sizes of features is crucial for the shape of the part to satisfy is function. Features are generally compared to a base feature, such as the initial protrusion, to avoid the generation of too many rules. The dimension rules do not act as a screening layer, unlike the shape rules, but do affect the systems confidence in classifying a part to a specific category. As a result features that fail the dimension rules are not rejected by the system, however the confidence level of the particular execution path is reduced to account for the unsuited feature dimensions.

3.4 Similarity Rules

These rules are used before or along side the placement rules and check for the similarity of features, whenever needed. Features are dubbed as similar if they are alike and are placed in similar positions with respect to the main protrusion. As an illustration, the gear cuts, while constructing a model of the gear category, must have the same dimensions for the gear to mesh properly with its mate. The recognition process would first identify the gear cuts from the shape rules and then group them based on the results from these rules. Presence of different groups of gear cuts would cause the recognition process to follow independent paths for each of the groups.

The similarity rules have a limited impact on the parameters that help in deciding whether the input CAD model belongs to a particular category. They however play a larger role in association with the placement rules, since similar features in a group must be placed at certain positions relative to each other. These rules apply only to a few categories of objects where one or more such groups of features must be present in the model for it to be

classified as a member of the category. In case of the external spur gear, the similar gear cuts are grouped and inspected to see if they form a circle and also if they are located at exactly the same distance from each other.

3.5 Placement and Orientation Rules

After the screening of features of suitable shape and sizes, the position of the features in the part needs to be ascertained in order to understand the geometry of the part. The placement and orientation rules check for the position of the feature in the part. The specific position of feature may depend on the position of other features in the part. These rules are thus computationally expensive and should be modeled with proper data structures to ensure the best possible running times.

Examples of placement rules would be the location of the gear cuts in a gear model. In case of the external spur gear, the gear cuts must be placed on the cylindrical surfaces of the protrusion; while for an internal spur gear model, they must lie on the cylindrical surfaces of a hole in the protrusion. For the straight bevel gear, the gear cuts should be located on the conical surfaces of the protrusion and they can be placed on any of the protrusion planes of a rack gear model.

While the aforesaid rules check for position of a feature relative to the main protrusion of the gear, other placement rules check for position of a feature with respect to other features in the model. This ensures that the gear cuts are placed within a specific distance from each other. Thus for the spur (external and internal) and bevel gears, the gear cuts must form a circle and for the rack gear they must be in a straight line.

Likewise to the placement rules, the orientation rules make sure that a feature is oriented in the right direction with respect to other features in the part. An example of this rule would be that of the orientation of the hole in spur and straight bevel gear models. The axis of the hole should be parallel to the axis of the protrusion cylinder in an external spur gear model and to the axis of the protrusion cone in a straight bevel gear model. The orientation requirements of the gear cuts in the external spur gear could be specified by the direction of the normal vectors to the three surfaces of the gear cut. These vectors must be perpendicular to the axis of the protrusion cylinder.

3.6 Chapter Summary

The rules for the shape and size of features and certain relationships among them are classified into four broad groups; the shape rules, the dimension rules, the similarity rules and the placement and orientation rules. These rules have been discussed in this chapter accompanied by selected examples. In the following chapter the recognition system developed in this research along with the flow of control is explained. A potential application of the recognition system as a database query processor is also presented.

4 RECOGNITION SYSTEM AND FLOW OF CONTROL

4.1 Introduction

This chapter describes in detail a system developed to demonstrate the working and function of a model recognition system. The system recognizes complete as well incomplete CAD models of the category *Gear* with four sub-categories; the *External Spur Gear*, the *Internal Spur Gear*, the *Straight Bevel Gear* and the *Rack Gear*. The system generates a recognition parameter termed as the *confidence level*, which indicates the point to which the system considers the input model to be a member of the sub-category. The system also displays a *level of completeness* pointing to the levels and features which are completed or missing from the construction tree of the input model when compared to the construction tree of the category. The output is then utilized to access data from a database and thus presents an application of the recognition process as a multimedia database query processor.

4.2 Input Files

The choice of the type of input files depends largely on the data required to carry out the recognition process. The system currently developed, requires model information in DSG format with data clearly portraying the parent-child relationships among features. It relies on

external sources for carrying out the feature extraction and representation of the input model in the DSG format. To concentrate on the research objectives, the feature extraction procedure is replaced by the use of a *neutral* and an *inf* file format developed by Pro/ENGINEER, with the assumption that the user develops the input model in the DSG format.

The neutral file stores information, in ascii format, concerning the different features developed during the generation of the model. The information is saved as objects of structures for different part entities such as features, surfaces, edges, etc. Appendix B describes the necessary elements of the neutral file format in detail. The inf file format stores information concerning the parent-child relationships among features. It references features by a unique id assigned to them during model generation by Pro/ENGINEER and for each feature displays the ids of its parents and its children.

4.3 Recognition System

The system stores a representation of the construction tree of the categories it recognizes. A tree data structure is used to describe the construction tree of a category in which each node, excluding the root node can have any number of parents and children, thereby, similar to a directed acyclic graph. Along with this tree data structure are functions that correspond to the specific rules that must be sufficiently satisfied to classify the input model. These functions generally have input parameters as one or more features and output as either a Boolean value or a real number between 0 and 100. For a Boolean output, 0 signifies that the input feature or features have failed the rule while 1 signifies that they have

passed the rule. For a real number output, a value of 0 implies that the input parameters have completely failed to satisfy the requirements of the rule and a value of 100 implies the input parameters are in complete agreement with the requirements of the rule. Any value in between would indicate the point to which the parameters satisfy the constraints set by the rule. A threshold value is chosen for some rules which denotes the minimum level of compliance of the input parameters with the constraints represented by the rule, that must be attained for proceeding with the feature or features to the next stage.

The system code has an object-oriented design developed in C++. The categories are represented as classes in the code and the rules are member functions of these classes. A base class called *Gear* corresponds to the gear category and derived classes *ExternalSpurGear*, *InternalSpurGear*, *StraightBevelGear* and *RackGear* correspond to the four sub-categories whose members are recognized by the system.

4.4 Recognition Process

The process starts with the parsing of the input files. The relevant data in the input files is used to initialize objects of different classes such as the *Feature* class, the *Surface* class and the *Edge* class. These objects are used to create a construction tree representation of the input model with the main protrusion at the root node. The construction tree is then compared to that of a gear sub-category to obtain a confidence level and a level of completeness factor.

The protrusion in the input model is first checked for shape rules. If the protrusion successfully passes the shape rules of the gear sub-category, the confidence level is suitably

raised and the input model is said to have completely passed the first level as a member of that particular category. For the external spur gear category, the model in Figure 24 is said to have passed level 1 and has a confidence level of 20.0%. For the internal spur gear category, there is no precise definition of the shape of the protrusion and so the models in Figure 36 and Figure 37 are termed as level 1 complete and assigned a confidence level of 10.0%. In case of the rack gear category, the rectangular prism model displayed in Figure 48 is level 1 complete with a confidence level of 20.0%. Similarly the cone shown in Figure 56 is level 1 complete for the straight bevel category with a confidence level of 20.0%.

If the input protrusion is not passed, the system invokes a pattern recognizer to recognize the input model. The pattern recognizer is initiated whenever an intermediate level fails during the recognition process. It tests for the presence of a specific shape pattern that defines a particular category. For instance, the model in Figure 31 fails to pass the first level and so is passed to the pattern recognizer, which ascribes a confidence level of 51.26% to the model. The pattern recognizer in the external spur gear category searches for a group of gear cuts with the same shape, size and orientation, placed within reasonable distances from each other and forming a circle. The recognizer also searches for a hole with the right orientation and diameter. Similar pattern recognizers have been developed for the internal spur gear, straight bevel gear and the rack gear category. The pattern recognizer is only able to recognize complete models of a particular category and cannot determine the level of completeness of the input model, thereby limiting its use.

If the first level is complete the system begins a search for features that make up the next level in the particular category's construction tree. For instance, the module for the external spur gear category would search the input model's construction tree for gear cuts

and a hole that are of the right shape, size and orientation and placed in the correct position, relative to the main protrusion. The confidence level is increased if rules are successfully passed. Figure 25 displays an image of a model tested for the external spur gear category. The presence of a hole with the right diameter and orientation results in an increase of confidence level to 30.0%, and the input model is termed as level 1 complete and level 2 partially complete. Besides testing for the presence of appropriate features, the system ensures that there are no features interfering with the surfaces needed for the part to perform its functions. In the gear category, the gear teeth surfaces are the surfaces needed for the gear to mesh with its mate and hence carry out its function. The presence of any features on these surfaces brings about a likewise reduction in the final confidence level of the input model.

In all the gear sub-categories described in this system, the gear cuts have to be similar in shape, size and in the position where they are located in the part. As a result, the gear cuts found in the construction tree of the input model are first grouped by the similarity rules and then checked against the placement rules. The presence of a single group of gear cuts with a sufficient number of gear cuts that pass all the placement rules increases the confidence level of the input model. If all the rules are satisfactorily passed, the input model is said to be complete in the last level of the category's construction tree. Figures 26, 27, 28 and 29 show images of models that are level 2 complete for the external spur gear category. Figures 30, 32, 33, 34 and 35 show the effect of incorrect feature shapes and sizes on the confidence level of the system.

The entire recognition process can be viewed as a traversal along a graph to find the path with the largest confidence level. The traversal strategy is thus depth first search where the recognition process stops when it finds a path with a 100% confidence level or when all

the paths have been exhausted. In the latter case, the highest level of completeness and confidence level among all the paths is considered as the output for the input model. Figure 43 provides an illustration of the depth first search traversal. The model in this figure is checked for membership to the internal spur gear category. The presence of two holes causes the system to follow two independent paths and return the maximum acquired confidence level. The more important rules used for the recognition of the four gear sub-categories are described in Tables 1, 2, 3 and 4 below. Figure 22 below graphically demonstrates the flow of control of the system.

4.5 Confidence Level Calculation

To understand the procedure for calculation of the confidence level, an example of a straight bevel gear model shown in Figure 63 is considered. The example is used to give a broad idea of the system's flow of control and the usage of the recognition rules. The model in Figure 63 is a level 3 complete straight bevel gear, with a confidence level of 98.77%.

As mentioned earlier, the system starts with the parsing of the input files, followed by the initialization of the various data objects and creation of the input model's construction tree. The protrusion at the root node of the construction tree is then checked against the shape rules for a cone. Since the shape of the protrusion satisfies all the shape rules, the model is assigned a confidence level of 20%.

Check is then made for the level 2 features, which in this case is a cut. The cuts in the model construction tree are tested against the shape rules of a frustum cut that is needed to generate a frustum. A single cut in the input model construction tree passes these shape rules,

and following that is subsequently checked for dimension, placement and orientation rules. The cut feature passes all the rules and as a result the system increases the confidence level of the model to 35%.

The sub-tree emanating from the frustum cut feature is checked for level 3 features. The system first looks for all the slots in the sub-tree, and in this case finds a single slot that is checked against the shape dimension, placement and orientation rules of a hole. The feature successfully passes all the requirements and consequently the system increases the confidence level of the model to 45%. Next, the cut features in the sub-tree are tested against the shape rules of a gear cut feature. At this stage, the cuts fail to completely satisfy the rules since the angles between the side surfaces and the middle surfaces of the cuts are not the optimal value. The system accordingly increases the confidence level of the model by an amount determined by the deviation of the angle from the optimum and hence the confidence level is increased to 53.77%.

Following the shape rules, the gear cuts pass the dimension, orientation and some of the placement rules, and then are grouped by the similarity rules. Presence of sufficient number of gear cuts in a group increases its confidence level. Here, all the gear cuts are grouped in a single set since they are similar in shape, size, orientation and location in the part. The confidence level is thus increased to 93.77%. A check is now made to ensure that no undesired features intersect with the conical surfaces in between the gear cuts, i.e. the surfaces making up the gear teeth. Lastly the group is checked for circularity and distances between the group members. These placement rules are also met satisfactorily, and the system assigns a final confidence level of 98.77% to the model.

Table 1 External Spur Gear Rules

Rule	Description
checkLevelOne	Checks for the presence of a protrusion in the first level of the input model's construction tree.
checkCylinderShape	Checks if an input feature satisfies the shape requirements of a cylinder.
checkLevelTwo	Checks for the presence of cuts and slots in the input model's construction tree.
checkHoleShape	Checks if an input feature satisfies the shape requirements of a hole.
checkHoleDimension	Checks if the dimensions of a hole relative to the protrusion are within acceptable limits.
checkHoleOrientation	Checks if a hole is oriented in the proper direction with respect to the protrusion.
checkHolePlacement	Checks if a hole is placed in the proper position with respect to the protrusion.
checkGearCutShape	Checks if an input feature satisfies the shape requirements of a gear cut.
checkGearCutDimension	Checks if the dimensions of a gear cut relative to the protrusion are within acceptable limits.
checkGearCutOrientation	Checks if a gear cut is oriented in the proper direction with respect to the protrusion.
checkGearCutPlacement	Checks if a gear cut is placed in the proper position with respect to the protrusion.
checkForSimilarGearCuts	Creates sets of similar gear cuts by grouping them based on their shape, size and location in the part.
checkForCircularity	Checks if a set of similar gear cuts form a complete circle, and are placed within certain distances of each other based on their shape and size.
checkForInterference	Checks if any unnecessary features intersect the surfaces which lie in between the gear cuts, and which form the functional surfaces of the gear.
checkForPattern	Invokes the pattern recognizer.

Table 2 Internal Spur Gear Rules

Rule	Description
checkLevelOne	Checks for the presence of a protrusion in the first level of the input model's construction tree.
checkLevelTwo	Checks for the presence of a slot in the input model's construction tree.
checkHoleShape	Checks if an input feature satisfies the shape requirements of a hole.
checkLevelThree	Checks for the presence of cuts in a sub-tree with a hole at its root node.
checkGearCutShape	Checks if an input feature satisfies the shape requirements of a gear cut.
checkGearCutDimension	Checks if the dimensions of a gear cut relative to the hole are within acceptable limits.
checkGearCutOrientation	Checks if a gear cut is oriented in the proper direction with respect to the hole.
checkGearCutPlacement	Checks if a gear cut is placed in the proper position with respect to the hole.
checkForSimilarGearCuts	Creates sets of similar gear cuts by grouping them based on their shape, size and location in the part.
checkForCircularity	Checks if a set of similar gear cuts form a complete circle, and are placed within certain distances of each other based on their shape and size.
checkForInterference	Checks if any unnecessary features intersect the surfaces which lie in between the gear cuts, and which form the functional surfaces of the gear.
checkForPattern	Invokes the pattern recognizer.

Table 3 Rack Gear Rules

Rule	Description
checkLevelOne	Checks for the presence of a protrusion in the first level of the input model's construction tree.
checkRectangularPrismShape	Checks if an input feature satisfies the shape requirements of a rectangular prism.
checkLevelTwo	Checks for the presence of cuts in the input model's construction tree.
checkGearCutShape	Checks if an input feature satisfies the shape requirements of a gear cut.
checkGearCutDimension	Checks if the dimensions of a gear cut relative to the protrusion are within acceptable limits.
checkGearCutOrientation	Checks if a gear cut is oriented in the proper direction with respect to the protrusion.
checkGearCutPlacement	Checks if a gear cut is placed in the proper position with respect to the protrusion.
checkForSimilarGearCuts	Creates sets of similar gear cuts by grouping them based on their shape, size and location in the part.
checkForLinearity	Checks if a set of similar gear cuts are in line, and are placed within certain distances of each other based on their shape and size.
checkForInterference	Checks if any unnecessary features intersect the surfaces which lie in between the gear cuts, and which form the functional surfaces of the gear.
checkForPattern	Invokes the pattern recognizer.

Table 4 Straight Bevel Gear Rules

Rule	Description
checkLevelOne	Checks for the presence of a protrusion in the first level of the input model's construction tree.
checkConeShape	Checks if an input feature satisfies the shape requirements of a cone.
checkLevelTwo	Checks for the presence of cuts in the input model's construction tree.
checkFrustumCutShape	Checks if an input feature satisfies the shape requirements of a frustum cut.
checkFrustumCutDimension	Checks if the dimensions of a frustum cut relative to the protrusion are within acceptable limits.
checkFrustumCutOrientation	Checks if a frustum cut is oriented in the proper direction with respect to the protrusion.
checkFrustumCutPlacement	Checks if a frustum cut is placed in the proper position with respect to the protrusion.
checkLevelThree	Checks for the presence of cuts and slots in a sub-tree with a frustum cut at its root node.
checkHoleShape	Checks if an input feature satisfies the shape requirements of a hole.
checkHoleDimension	Checks if the dimensions of a hole relative to the protrusion and the frustum cut are within acceptable limits.
checkHoleOrientation	Checks if a hole is oriented in the proper direction with respect to the protrusion and the frustum cut.
checkHolePlacement	Checks if a hole is placed in the proper position with respect to the protrusion and the frustum cut.
checkGearCutShape	Checks if an input feature satisfies the shape requirements of a gear cut.
checkGearCutDimension	Checks if the dimensions of a gear cut relative to the protrusion and the frustum cut are within acceptable limits.
checkGearCutOrientation	Checks if a gear cut is oriented in the proper direction with respect to the protrusion and the frustum cut.

Table 4 (continued)

Rule	Description
checkGearCutPlacement	Checks if a gear cut is placed in the proper position with respect to the protrusion and the frustum cut.
checkForSimilarGearCuts	Creates sets of similar gear cuts by grouping them based on their shape, size and location in the part.
checkForCircularity	Checks if a set of similar gear cuts form a complete circle, and are placed within certain distances of each other based on their shape and size.
checkForInterference	Checks if any unnecessary features intersect the surfaces which lie in between the gear cuts, and which form the functional surfaces of the gear.
checkForPattern	Invokes the pattern recognizer.

4.6 Application

The code written and the test cases generated are made available in Appendix A below. The package created provides the user with a user interface (Figure 23) to input the CAD files and obtain an output. The user-interface is developed using RapidApp API and OpenGL to display the input model. The interface uses a user-defined threshold value to classify the input model to a particular category. If the confidence level of the input model exceeds the threshold value, the input model is accepted as a member of the category, else it is rejected. The user can utilize the system as query processor for a multimedia database wherein the input model is used as the query for finding out information about its category. Thus, if the confidence level of the input model surpasses the threshold value, the model is classified as a member of the category and information regarding the category is provided to the user from the database.

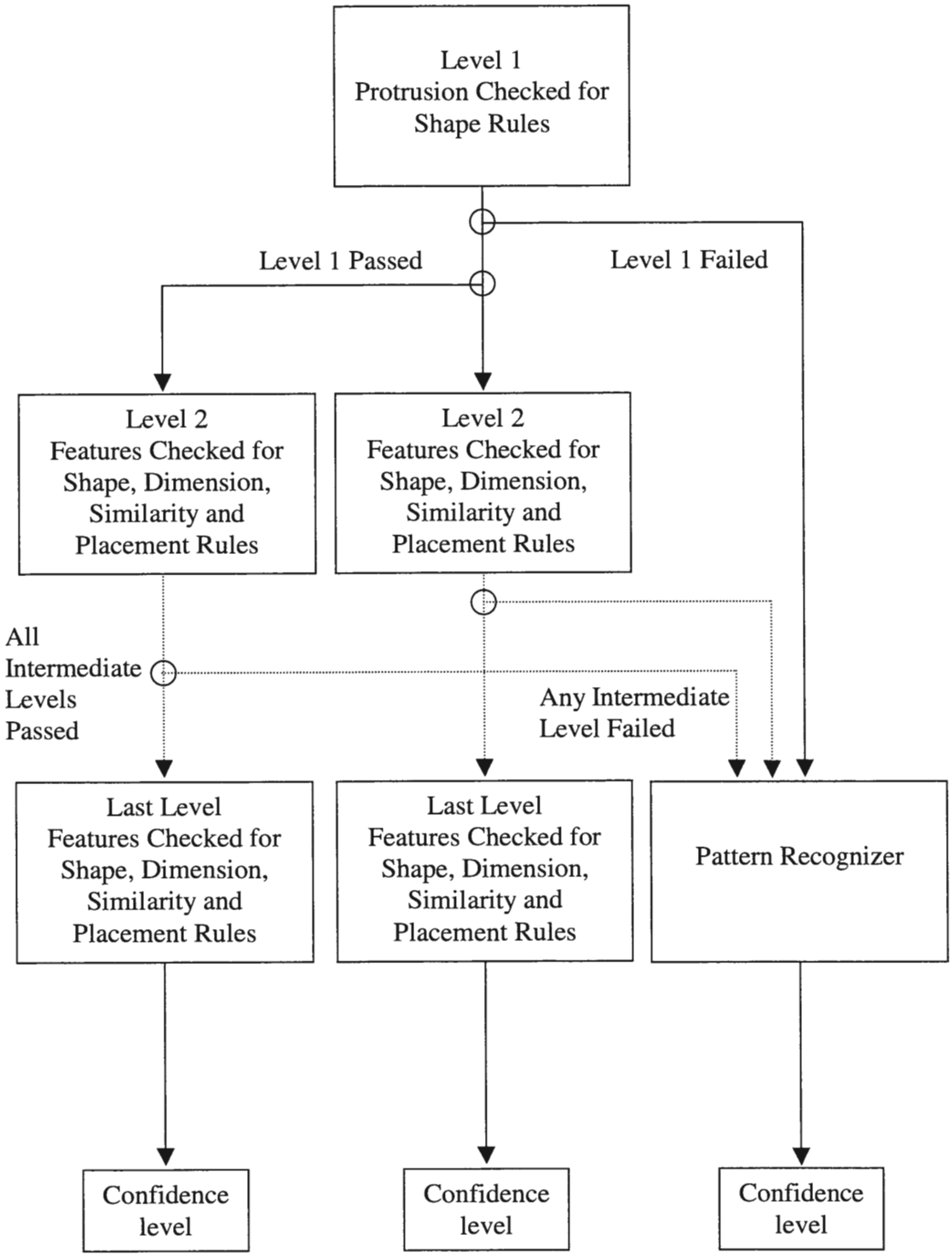


Figure 22 Flow of Control

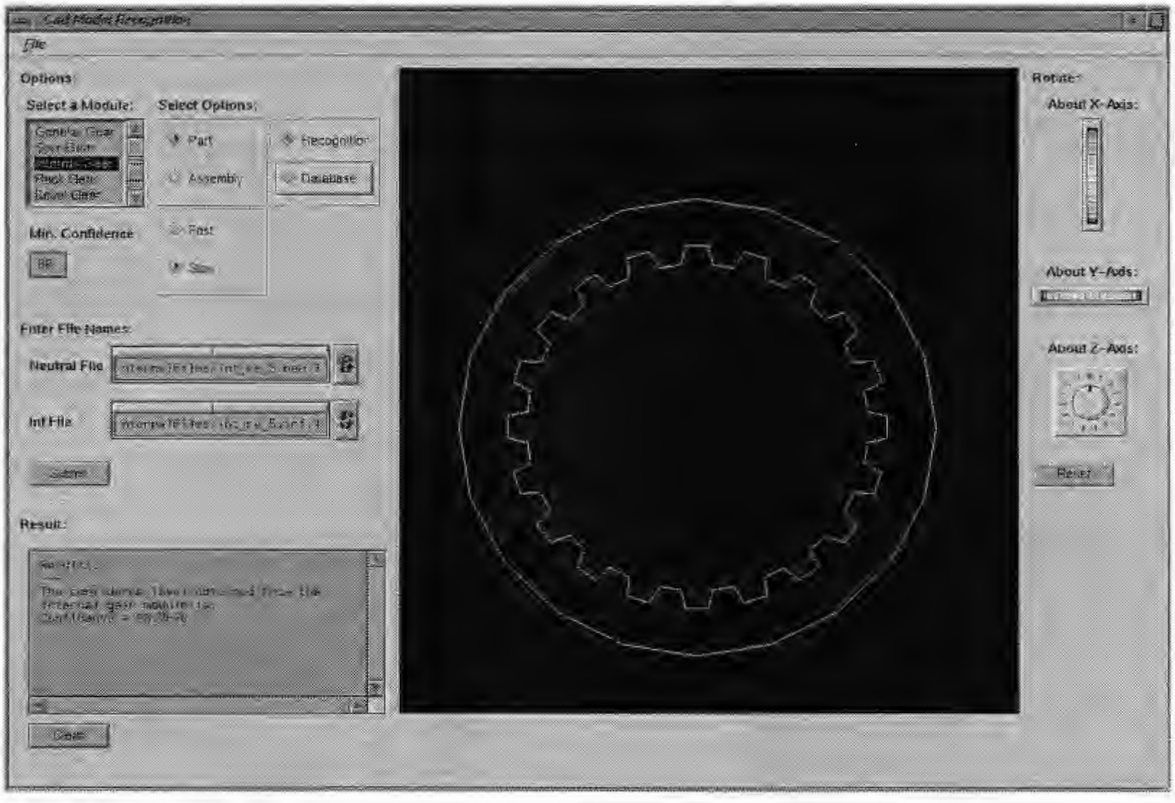


Figure 23 The User-Interface with a sample input model displayed

4.7 Chapter Summary

The chapter starts off with the description of the input files followed by an explanation of the recognition system and the flow of control. The process of calculation of the confidence level is also discussed here, with corresponding examples referred to Chapter 5. In Chapter 5 a small set of the test cases generated are displayed with their confidence levels that were obtained from the system. The test cases cover all the four gear sub-categories, the external spur gear, the internal spur gear, the rack gear and the straight bevel gear.

5 TEST CASES

5.1 Test Cases

In order to appropriately put a claim on the competence and capabilities of the system, a logical set of test cases have been developed. The selection criterion of this set has been such that a given test case must assess the effect of at least one of the rules described earlier. The assessment can be done by being familiar with the weight of the rules for which the test case was generated and by observing the change in the overall confidence level obtained from the output of the recognition system. Besides checking for the fact that the system is able to provide a reasonably correct confidence level for the test case, the system's competence should be noted by its ability to determine the user's intent by classifying the test case in the appropriate categories.

A few of the test cases generated have been presented below along with the summary of the output from the recognition system. Models in figures 24 – 35 have been tested for the external spur gear module, while those in figures 36 – 47 have been used to analyze the internal spur gear module. Following these are figures of models used to examine the rack gear component (Figures 48 – 55) and the bevel gear component (Figures 56 - 67) of the system.

The model in Figure 24 is level 1 complete with a confidence level of 20.0%. The confidence level of 20.0% is allotted since the protrusion is passed as a cylinder.

Figure 24 External Spur Gear 1

The model in Figure 25 is level 1 complete and level 2 partially complete with a confidence level of 30.0%. The confidence level of this model is more than that of the model in Figure 24 due to the presence of a suitable hole.

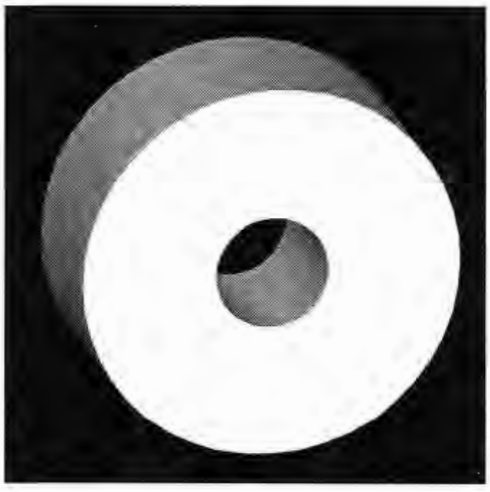


Figure 25 External Spur Gear 2

The model in Figure 26 is level 1 and 2 complete with a confidence level of 96.81%. The confidence level is little less than 100.0% since the shape of the gear cuts is not optimal.

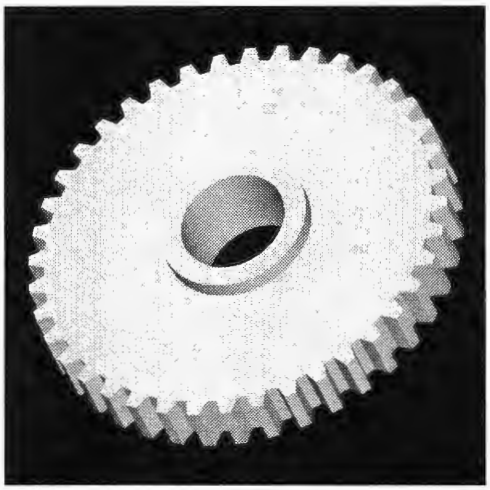


Figure 26 External Spur Gear 3

The model in Figure 27 is level 1 and 2 complete with a confidence level of 96.15%. The confidence level is little less than 100.0% since the shape of the gear cuts is not optimal.

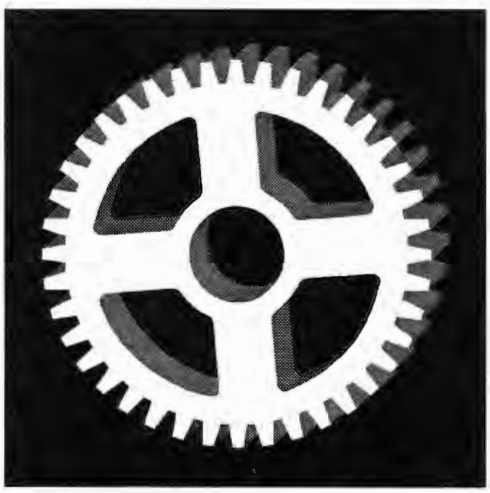


Figure 27 External Spur Gear 4

The model in Figure 28 is level 1 and 2 complete with a confidence level of 95.13%. Since the shape of the gear cuts is further away from the optimal as compared to the shape of the gear cuts in the model in Figure 27, the confidence level is comparatively smaller.

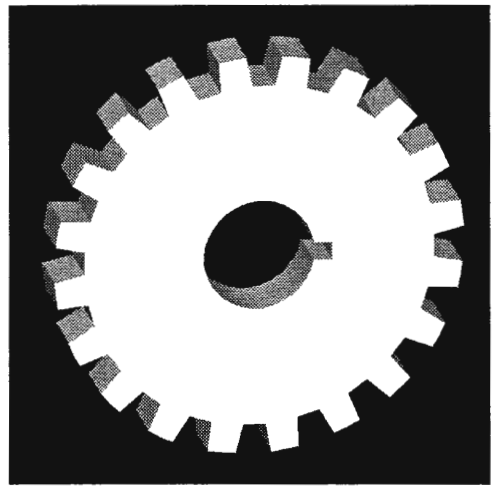


Figure 28 External Spur Gear 5

The model in Figure 29 is level 1 and 2 complete with a confidence level of 95.95%. The confidence level is less than 100.0% since the shape of the gear cuts is not optimal.

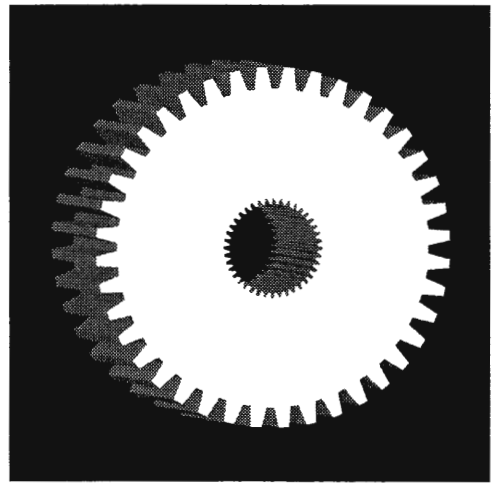


Figure 29 External Spur Gear 6

The model in Figure 30 is level 1 complete and level 2 partially complete with a confidence level of 30.0%. The shape of the cuts in the model is outside the acceptable limits and hence the cuts are rejected.

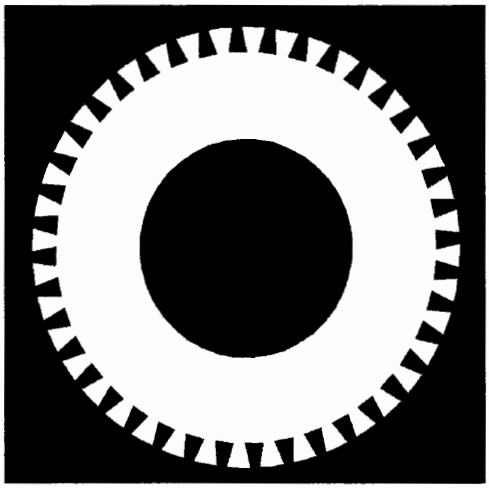


Figure 30 External Spur Gear 7

The model in Figure 31 is all levels incomplete with a confidence level of 51.26% allotted by the pattern recognizer.

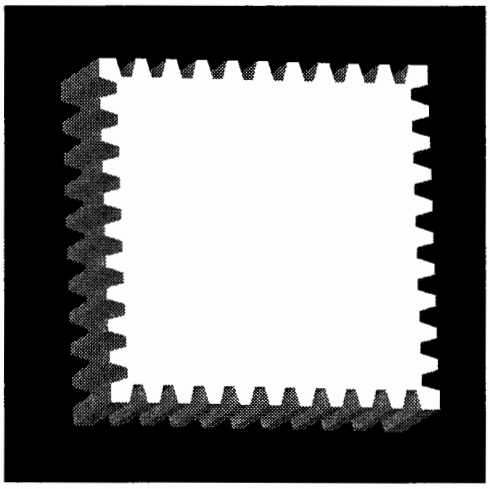


Figure 31 External Spur Gear 8

The model in Figure 32 is level 1 and 2 complete with a confidence level of 92.21%. The confidence level is reduced since the hole fails the dimension rules.

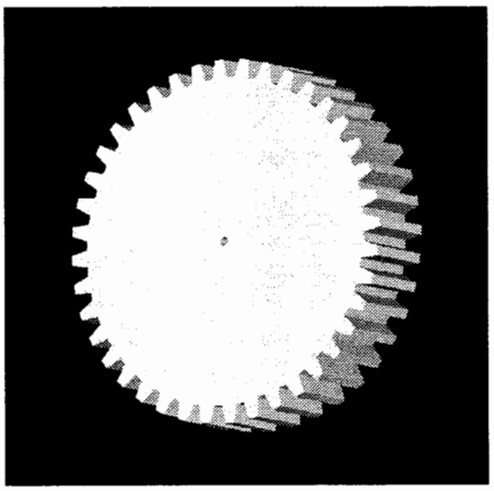


Figure 32 External Spur Gear 9

The model in Figure 33 is level 1 and 2 complete with a confidence level of 92.72%. The confidence level is reduced since the hole fails the dimension rules.

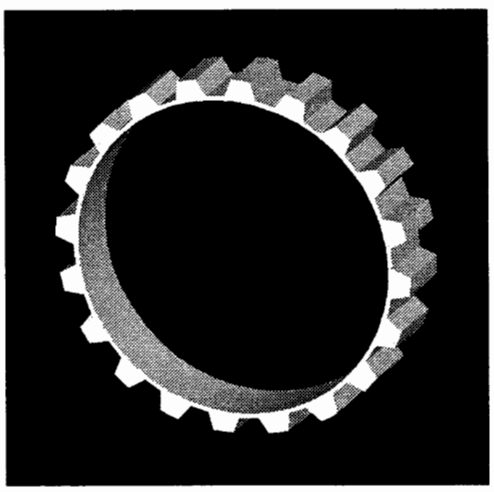


Figure 33 External Spur Gear 10

The model in Figure 34 is level 1 and 2 complete with a confidence level of 85.92%. The confidence level is reduced since the gear cuts fail the dimension rules.

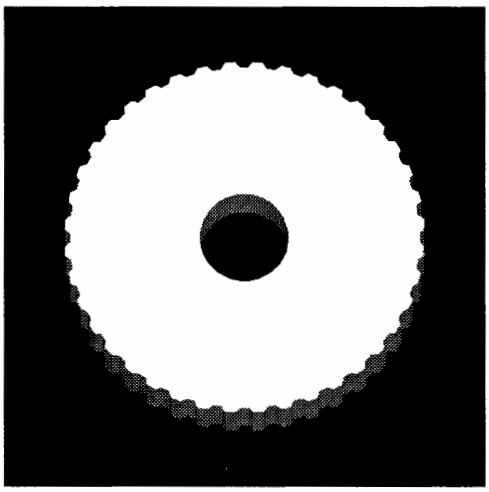


Figure 34 External Spur Gear 11

The model in Figure 35 is level 1 and 2 complete with a confidence level of 86.72%. The confidence level is reduced since the gear cuts fail the dimension rules.

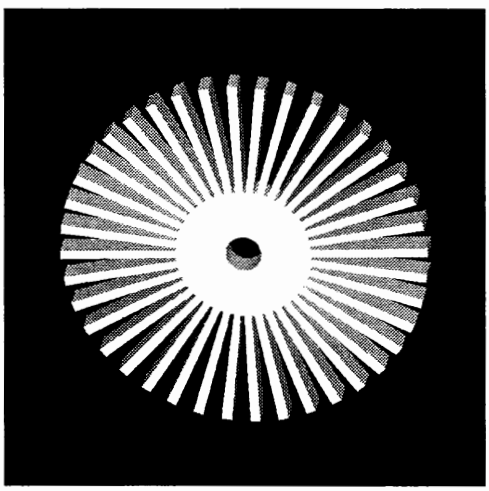


Figure 35 External Spur Gear 12

The model in Figure 36 is level 1 complete with a confidence level of 10.0% due to the presence of a protrusion.

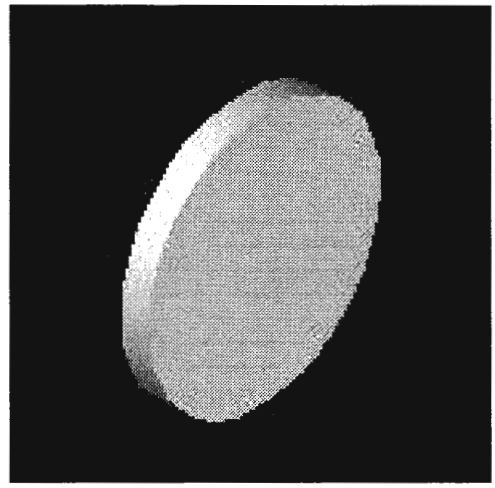


Figure 36 Internal Spur Gear 1

The model in Figure 37 is level 1 complete with a confidence level of 10.0% due to the presence of a protrusion.

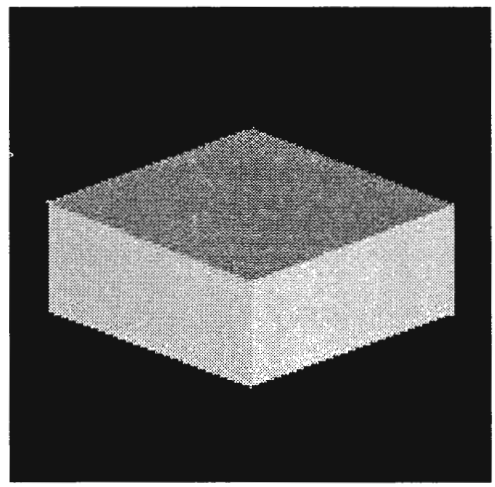


Figure 37 Internal Spur Gear 2

The model in Figure 38 is level 1 and 2 complete with a confidence level of 30.0%. The presence of a hole in the part results in a corresponding increase in the confidence level.

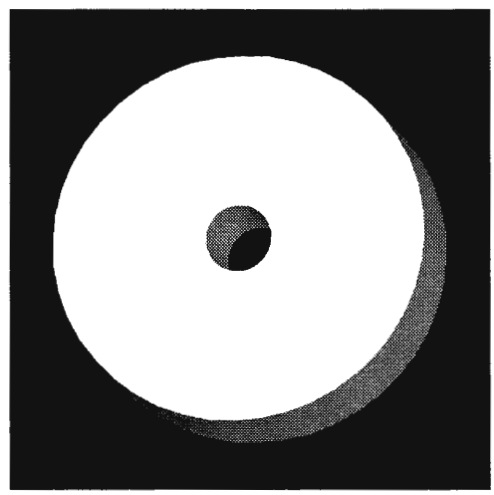


Figure 38 Internal Spur Gear 3

The model in Figure 39 is level 1 and 2 complete with a confidence level of 30.0%. Here too, the presence of a hole in the part results in a corresponding increase in the confidence level.

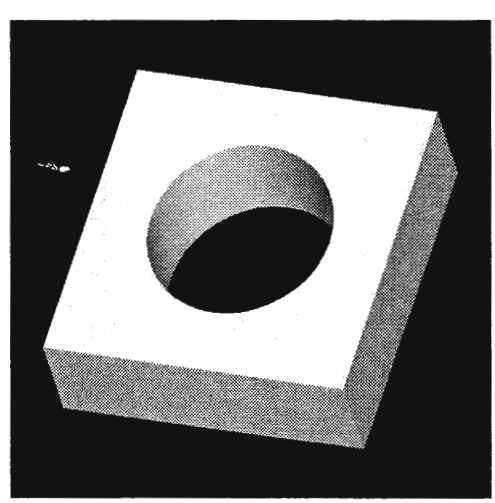


Figure 39 Internal Spur Gear 4

The model in Figure 40 is level 1, 2 and 3 complete with a confidence level of 93.85% since the gear cuts are not of the optimum shape.

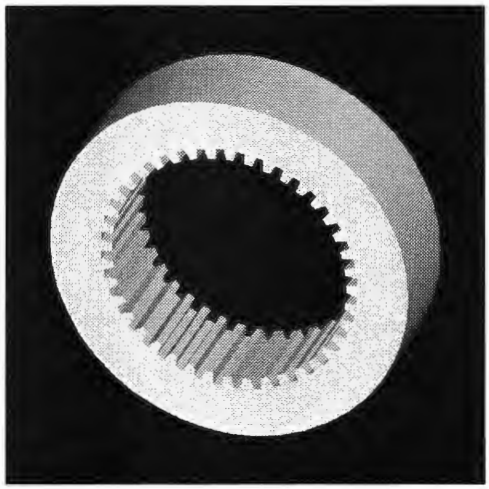


Figure 40 Internal Spur Gear 5

The model in Figure 41 is level 1, 2 and 3 complete with a confidence level of 96.67%. Again, the confidence level is slightly reduced since the gear cuts are not of the optimum shape.

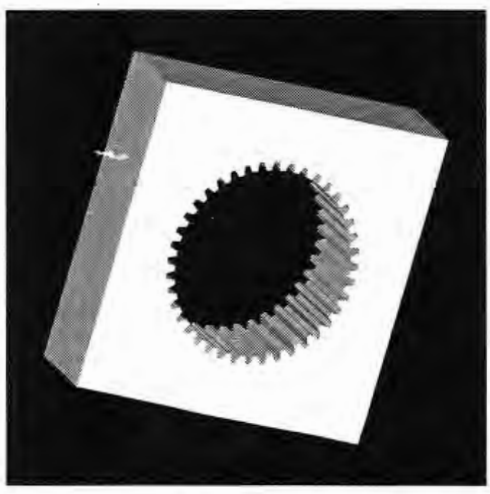


Figure 41 Internal Spur Gear 6

The model in Figure 42 is level 1, 2 and 3 complete. As compared to the shape of the gear cuts in the model in Figure 41, here, the shape of the gear cuts is further away from the optimum, resulting in a reduced confidence level of 92.62%.

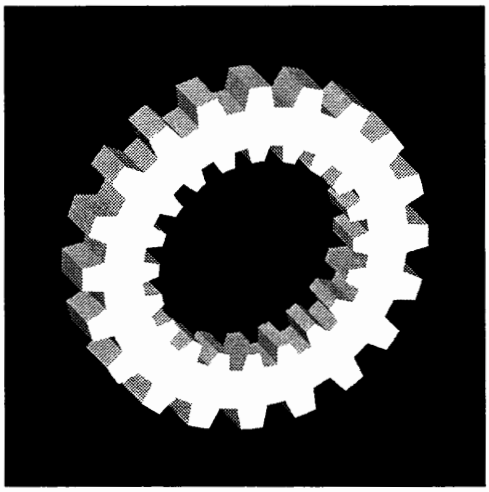


Figure 42 Internal Spur Gear 7

The model in Figure 43 is level 1, 2 and 3 complete with a confidence level of 94.60%, which is the maximum of the confidence levels obtained from the two holes.

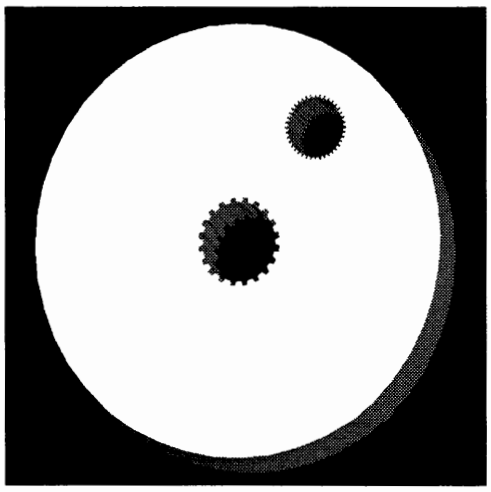


Figure 43 Internal Spur Gear 8

The model in Figure 44 is level 1, 2 and 3 complete with a confidence level of 87.31%. Since the depth of the gear cuts is too small as compared to the diameter of the hole, the gear cuts fail the dimension rules and thus the confidence level is decreased.

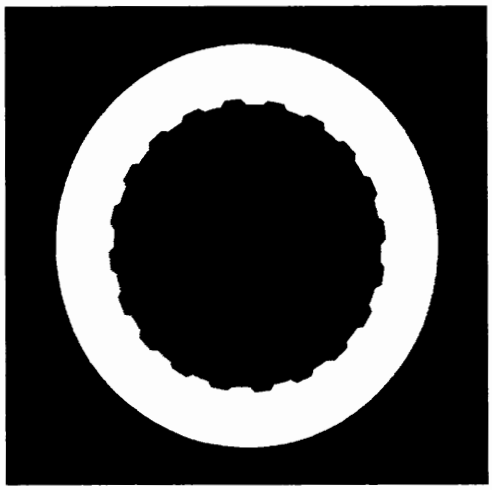


Figure 44 Internal Spur Gear 9

The model in Figure 45 is level 1, 2 and 3 complete. Here too, the gear cuts fail the dimension rules causing a decrease in the confidence level to 85.03%.

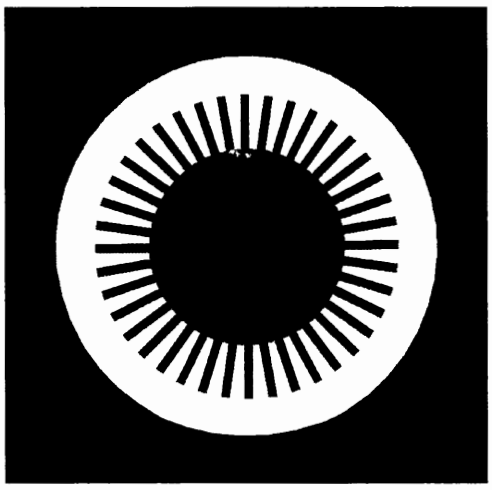


Figure 45 Internal Spur Gear 10

The model in Figure 46 is level 1 and 2 complete with a confidence level of 30.0%. The cuts in the model do not satisfy the shape rules for the gear cuts and so are discarded.

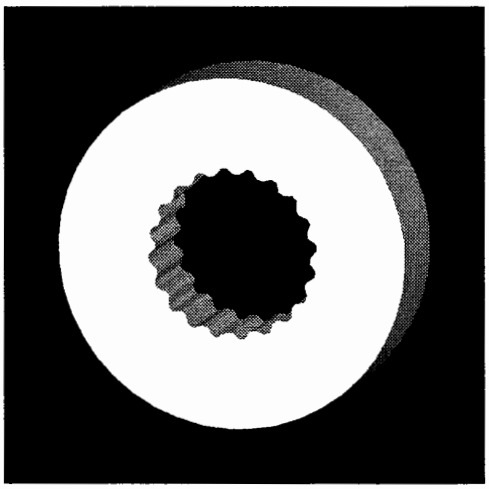


Figure 46 Internal Spur Gear 11

The model in Figure 47 is level 1 complete with a confidence level of 48.0%. Since no holes are present in the input model's construction tree, level 2 fails and the pattern recognizer is executed.

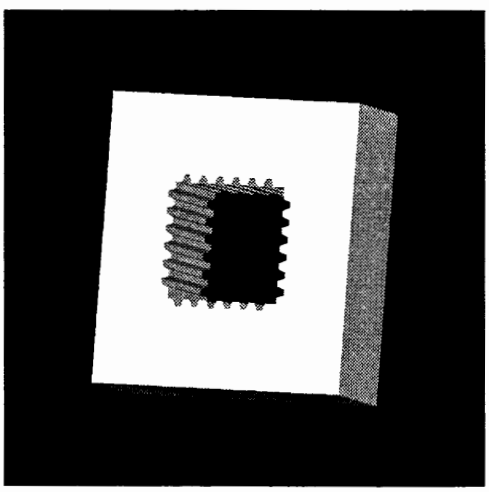


Figure 47 Internal Spur Gear 12

The model in Figure 48 is level 1 complete with a confidence level of 20.0% since the protrusion is passed as a rectangular prism.

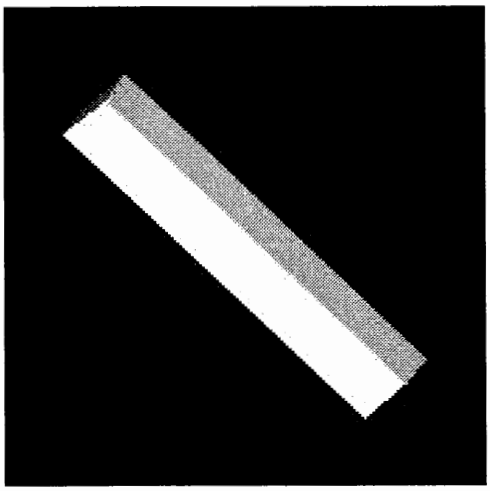


Figure 48 Rack Gear 1

The model in Figure 49 is level 1 and 2 complete with a confidence level of 97.09%. The shape of the gear cuts is not optimal and so the confidence level is a little less than 100.0%.

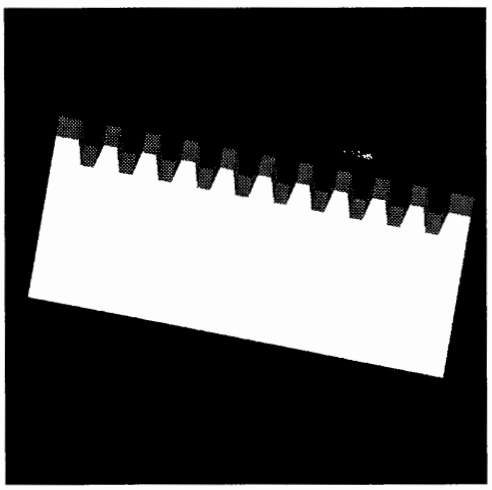


Figure 49 Rack Gear 2

The model in Figure 50 is level 1 and 2 complete with a confidence level of 98.21%, again due the fact that the gear cuts are not exactly of the best possible shape.

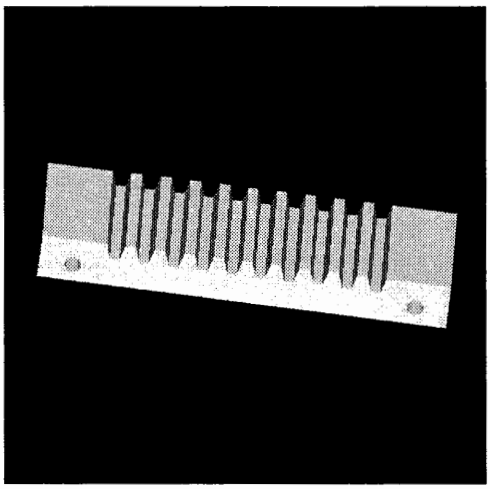


Figure 50 Rack Gear 3

The model in Figure 51 is level 1 and 2 complete with a confidence level of 83.52%. The presence of two different sets of gear cuts on the same surface of the rectangular prism causes the system to lower the confidence level.

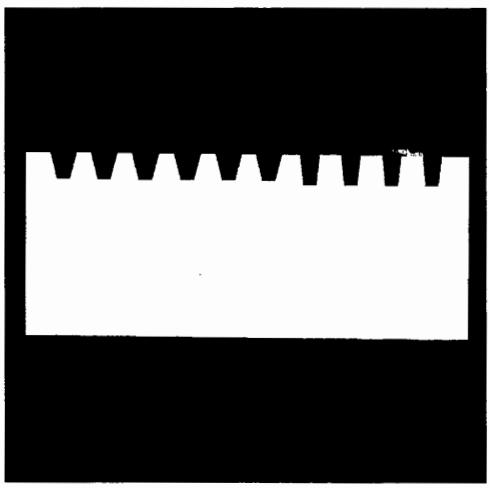


Figure 51 Rack Gear 4

The model in Figure 52 is level 1 and 2 complete with a confidence level of 97.09%. Since the gear cuts are present on two surfaces of the protrusion, the confidence level of the part is the maximum of the confidence levels obtained from each of the two sets of gear cuts.

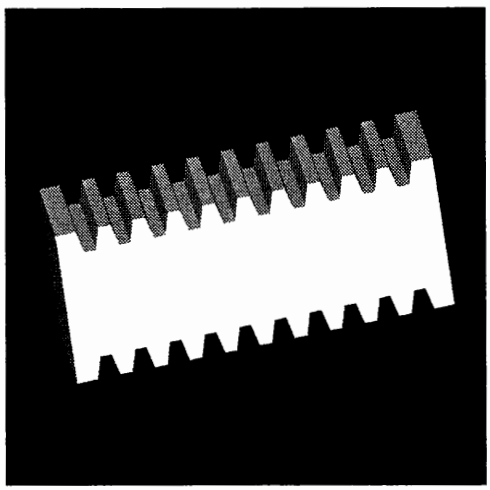


Figure 52 Rack Gear 5

The model in Figure 53 is level 1 and 2 complete with a confidence level of 94.85%, which is the maximum of the confidence levels obtained from the different sets of gear cuts.

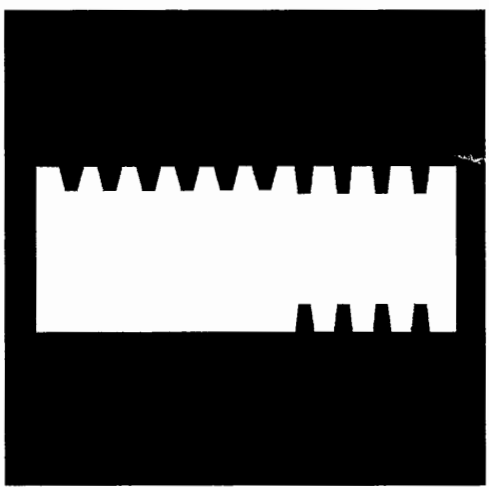


Figure 53 Rack Gear 6

The model in Figure 54 is level 1 and 2 complete with a confidence level of 93.19%. The confidence level is reduced since the gear cuts fail the dimension rules.

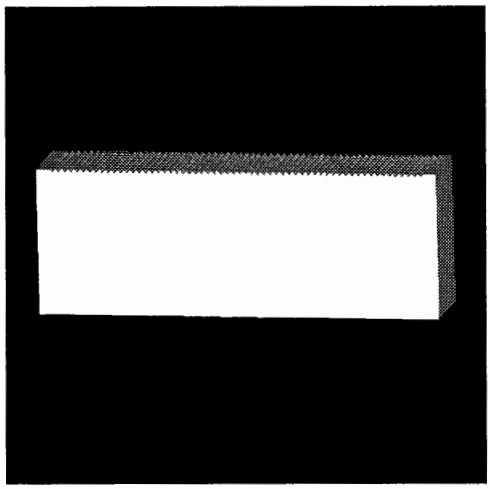


Figure 54 Rack Gear 7

The model in Figure 55 is level 1 and 2 complete with a confidence level of 86.62%. Here, since the depth of the gear cuts is too large as compared to the width of the protrusion, the gear cuts fail the dimension rules and the confidence level is reduced.

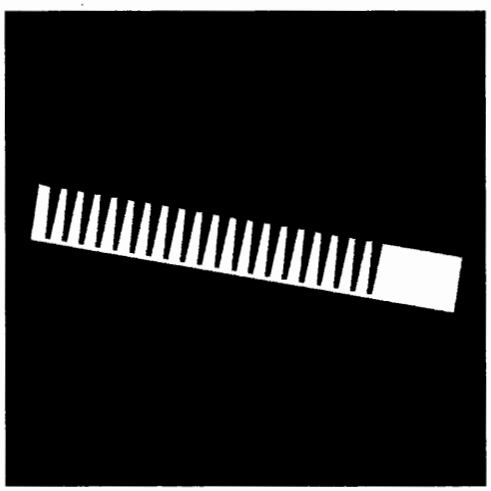


Figure 55 Rack Gear 8

The model in Figure 56 is level 1 complete with a confidence level of 20.0% since the protrusion is passed as a cone.

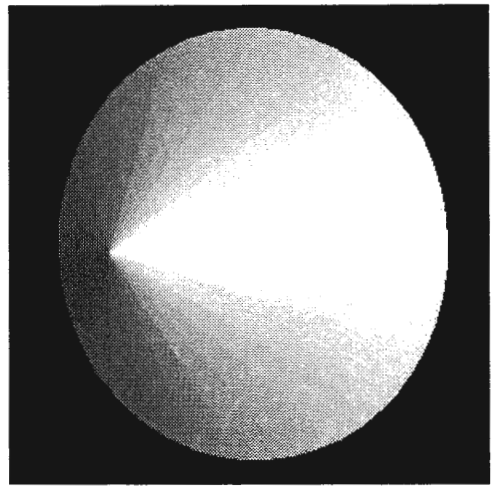


Figure 56 Straight Bevel Gear 1

The model in Figure 57 is level 1 and 2 complete with a confidence level of 35.0%. The presence of a suitable frustum cut in the model results in a corresponding increase in the confidence level.

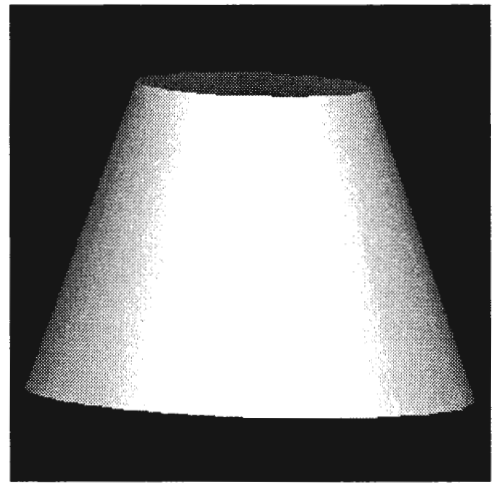


Figure 57 Straight Bevel Gear 2

The model in Figure 58 is level 1 and 2 complete and level 3 partially complete with a confidence level of 45.0% due the presence of a proper hole in addition to the conical protrusion and the frustum cut.

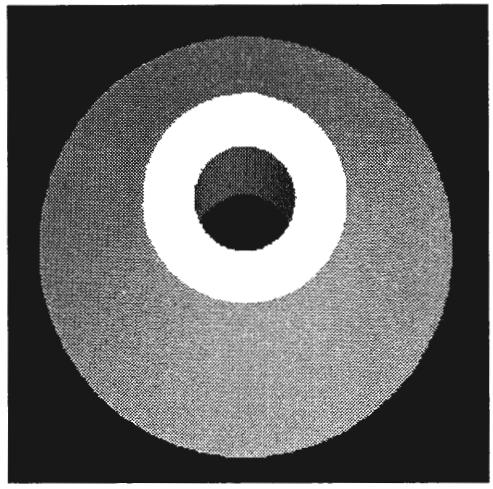


Figure 58 Straight Bevel Gear 3

The model in Figure 59 is level 1 and 2 complete and level 3 partially complete with an increase in the confidence level to 54.18% due to the addition of a suitable gear cut.

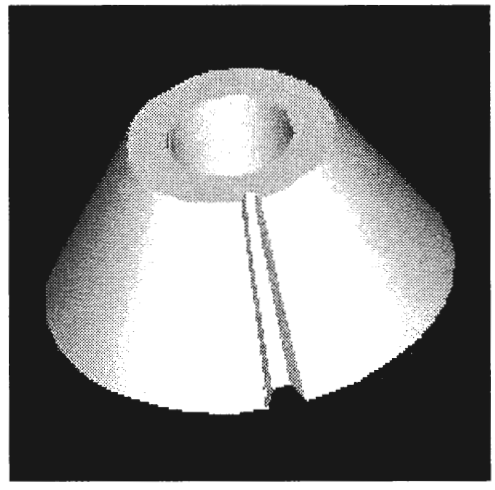


Figure 59 Straight Bevel Gear 4

The model in Figure 60 is level 1 and 2 complete and level 3 partially complete with a confidence level of 94.18% as a result of the gear cuts failing the placement rules.

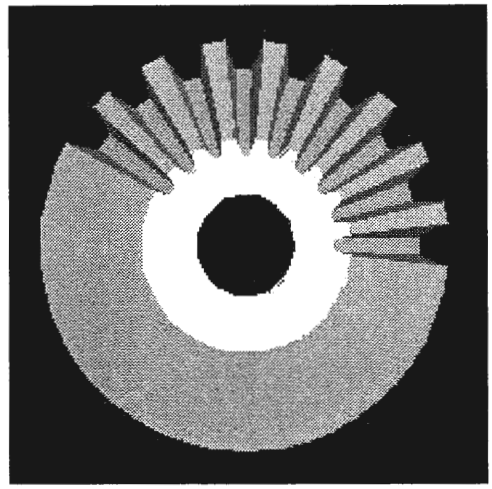


Figure 60 Straight Bevel Gear 5

The model in Figure 61 is level 1, 2 and 3 complete with a confidence level of 98.45%. The slight reduction in the confidence level is due to the fact that the shape of the gear cuts is not exactly the best possible.

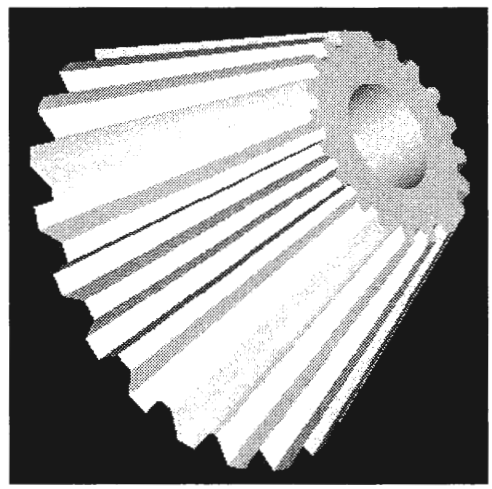


Figure 61 Straight Bevel Gear 6

The model in Figure 62 is level 1, 2 and 3 complete with a confidence level of 98.15%, again due to deviation in the shape of the gear cuts from the optimal.

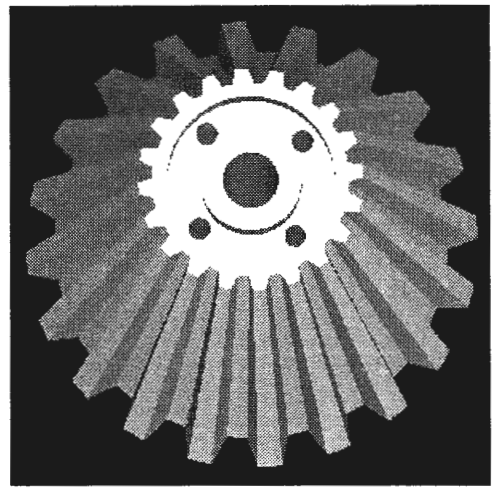


Figure 62 Straight Bevel Gear 7

The model in Figure 63 is level 1, 2 and 3 complete with a confidence level of 98.77%. In this case again, the shape rules for the gear cuts are not completely satisfied, resulting in a confidence level slightly lesser than 100.0%.

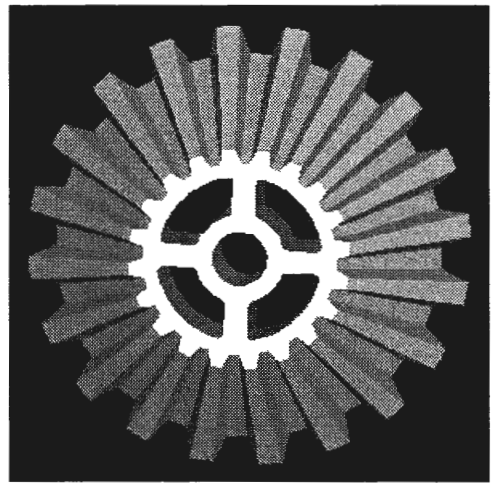


Figure 63 Straight Bevel Gear 8

The model in Figure 64 is level 1 and 2 complete and level 3 partially complete with a confidence level of 80.14%. The presence of two different sets of the gear cuts causes a substantial reduction in the confidence level.

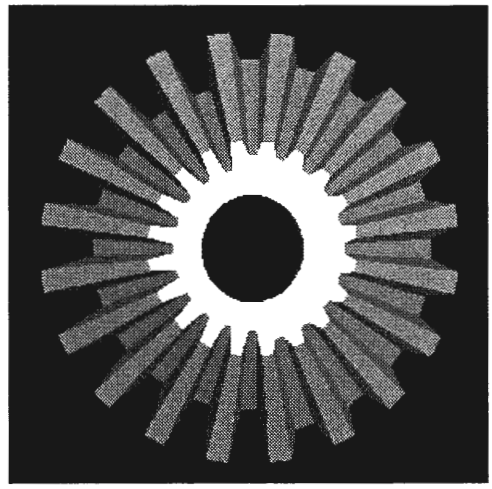


Figure 64 Straight Bevel Gear 9

The model in Figure 65 is level 1, 2 and 3 complete with a confidence level of 93.21%. The confidence level is decreased since the hole fails the dimension rules.

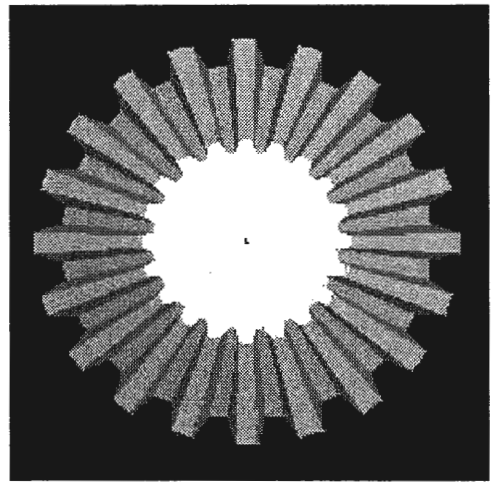


Figure 65 Straight Bevel Gear 10

The model in Figure 66 is level 1, 2 and 3 complete with a smaller confidence level of 93.20% since the hole in the model fails the dimension rules.

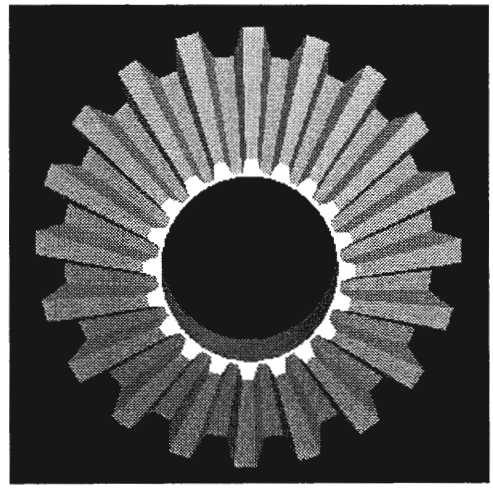


Figure 66 Straight Bevel Gear 11

The model in Figure 67 is level 1, 2 and 3 complete with a confidence level of 91.69%. As can be noticed, the gear cuts fail the dimension rules and the confidence level is appropriately decreased.

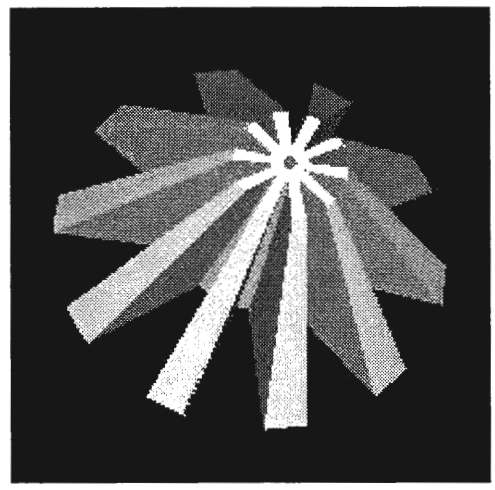


Figure 67 Straight Bevel Gear 12

5.2 Chapter Summary

Test cases are generated for each of the gear subcategories and are displayed in this chapter. The output obtained from the system for each of the test cases is explained in brief alongside the test cases. The subsequent chapter draws conclusions and provides suggestions for future directions of research in this area.

6 CONCLUSIONS

The research described here aimed in developing a system that can determine the user's objective by trying to recognize complete and incomplete CAD models. The central idea was to utilize object recognition techniques on a model construction tree, which denotes the sequence in which features were added in the model. A level in the model construction tree represents the level of completeness of the model. The input model's construction tree is compared with that of the object in question to obtain a confidence level and a level of completeness.

The recognition procedure defines the shapes and relationships of features using certain hints or rules. These rules are classified into four broad types and are discussed in detail for the gear category in this research. The flow of control in the recognition system is described along with an explanation of the calculation of the confidence level and the level of completeness. A few of the test cases developed have been presented with a brief note on the output for each of the test case. The system's application is currently limited to classifying input CAD models among of the gear sub-categories and using it to query a multimedia database.

Future work in the present area should concentrate on developing similar systems capable of recognizing models among a larger set of categories. Integration of these systems

with feature extraction modules capable of generating a unique DSG for an input CAD model would ensure their use in wide variety of applications such as the content-based retrieval systems mentioned earlier. In general, attention should be given to minimizing the design lead times by automating the model generation and retrieval processes using intelligent feature-based intent finders.

APPENDIX A SYSTEM CODE AND TEST FILES

The source code for the afore mentioned application, which include several header and class files, is on deposit with Dr. Ranga Narayanaswami, Assistant Professor of Industrial and Manufacturing Systems Engineering department at Iowa State University and may be obtained with the author's and his permission. The test files generated to test the system's competence and capabilities are also on deposit with Dr. Narayanaswami.

APPENDIX B NEUTRAL FILE FORMAT

A Neutral file is a text file containing topological information about parts and assemblies developed in Pro/ENGINEER. The information stored can easily be understood and used by different applications requiring information concerning features in a part.

Every line in the neutral file beginning with the character # is a comment which could be ignored. The lines not beginning with the character # store part data in the form of a *level number* which is an integer, followed by a *field* section which denotes the name of entity, and finally a *value* which stands for the value of the entity. The data in the *field* section is either one of the following:

1. The name of an object of a simple data type like integer, string, etc.
2. The name of an array.
3. The name of a structure data type
4. The name of a pointer to a structure data type

The *value* section can either be an integer, a floating point number or a string depending on the object represented by the *field* section. If the *field* section contains the name of an array, the *value* section would hold a series of array element values that could span over several lines. If the data in the *value* section is not a string, then that data is to be assigned to the *field* section. If the data is a string, it indicates that the *field* section is a pointer to a structure

whose attributes follow in the next few lines. The data may also be a special string, NULL, that indicates that the *field* section is a pointer whose value is null, i.e. the pointer has not been assigned. Lastly, the *value* section may be completely absent again indicating that the *field* section denotes a structure.

Part neutral files include the complete geometry and feature data needed to develop the model. The geometry is defined as a list of dimensions, features, surfaces, edges, etc created while generating the part model. Among these entities, the features, the surfaces and the edges of the part are of importance for this research and are described below.

Feature

Feature entities appear in sequential order in the neutral file starting with a comment line containing the name of the feature, which stands for the type of feature. The neutral file format classifies features as Protrusions, Cuts, Slots, Holes, etc. The feature name is followed by a unique string that denotes the id of the feature, which is used as a reference to the feature in the neutral file. Other elements of a feature include the dimensions created to define the shape and size of the features, the surfaces that were generated due to the addition of this feature, etc. Table 5 below describes in detail the data format needed to display all the information about a feature.

Table 5 Feature Entities

Field	Data Type	Description
Name of the feature	String	The name of the feature
id	Integer	The feature id
user_name	String	Name given by user
suppressed	Integer	Either 1 if feature is suppressed or 0 otherwise
dim_ids[]	Integer	Array of dimensions ids
surface_ids[]	Integer	Array of surface ids
edge_ids[]	Integer	Array of edge ids
misc_ids[]	Integer	Array of ids of miscellaneous items

Surface

The surface entities contain information regarding the surfaces that make up the part model. The data stored comprises of the boundary information of the surface, the edge-loops formed on the surface and the surface type. Each surface has been assigned a unique id with which the surface is referenced. The closed edge-loops which make up the surface boundaries are presented as an array of the loop data structure. The neutral file format classifies surfaces into several categories depending upon their shape. The different types of surfaces discussed here are the Planar, the Cylindrical and the Conical. Table 6 illustrates all the surface elements provided by the neutral file format.

Table 6 Surface Entities

Field	Data Type	Description
id	Integer	The surface id
uv_min[]	Double	The minimum u and v values
uv_max[]	Double	The maximum u and v values
xyz_min[]	Double	Minimum values of a box enclosing the surface
xyz_max[]	Double	Maximum values of a box enclosing the surface
orient	Integer	1 if surface normal points outside of part, else -1
loops[]	loop (structure)	Array of edge loops on the surface
surface_type	Integer	Constant indicating the type of surface
surface	surface (structure)	The surface structure

Planar Surface

The planar surface attributes consist of three mutually perpendicular unit vectors; two parallel to the plane (*e1* and *e2*), and the third one normal to the plane (*e3*). Also included is a point of origin lying on the plane to define it completely (Table 7). The different points on the plane can be obtained from the equation:

$$P(x, y, z) = (u * e1 + v * e2) + \text{origin}$$

where u and v are the variables in the two parametric directions.

Table 7 Plane Entities

Field	Data Type	Description
e1[]	Double	Unit vector in the u direction
e2[]	Double	Unit vector in the v direction
e3[]	Double	Unit vector in the normal direction
origin[]	Double	Origin point of the plane

Cylindrical Surface

The cylindrical surface attributes made available in the neutral file are a set of three mutually perpendicular unit vectors, a point of origin and the radius of the cylindrical surface. Among the set of perpendicular unit vectors, the vectors *e1* and *e2* are in the direction of the planar surfaces of the corresponding cylinder, while the vector *e3* is normal to these planar surfaces, parallel to the axis of the cylinder. Table 8 displays the above-mentioned fields and their description. The points on a cylindrical surface can be obtained from its attributes by using the following equation:

$$P(x, y, z) = (\text{radius} * (\text{cosine}(u) * e1 + \text{sine}(u) * e2)) + (v * e3) + \text{origin}$$

where u and v are the variables in the two parametric directions.

Table 8 Cylinder Entities

Field	Data Type	Description
e1[]	Double	Unit vector in the u direction
e2[]	Double	Unit vector in the v direction
e3[]	Double	Unit vector in the direction of the axis of the cylinder
origin[]	Double	Origin point of the cylinder
radius	Double	The radius of the cylinder

Conical Surface

The neutral file uses similar elements as the planar and the cylindrical surfaces to define a conical surface. The three mutually perpendicular vectors again have field names as *e1*, *e2* and *e3*, of which *e1* and *e2* are parallel to equivalent cone's base and *e3* is parallel to the axis of the cone. The point of origin of the surface and the cone's slant angle in radians form the other elements for this type of surface (Table 9). The equation below gives the points on a conical surface:

P (x, y, z) = v * tangent (alpha) * [cosine (u) * *e1* + sine (u) * *e2*] + v * *e3* + origin

where u and v are the variables in the two parametric directions.

Table 9 Cone Entities

Field	Data Type	Description
e1[]	Double	Unit vector in the u direction
e2[]	Double	Unit vector in the v direction
e3[]	Double	Unit vector in the direction of the axis of the cone
origin[]	Double	Origin point of the cone
alpha	Double	The slant angle of the cone

Edge

Information concerning the shape and size of an edge in the part model is described by this entity in the neutral file. Edges are identified by a unique id assigned to them when they are created. The edge entity maintains the id of the edge, along with the ids of the surfaces sharing this edge. An edge is classified as a Line, an Arc or a Spline, depending upon the shape of the curve. A constant signifying the type of curve is stored as an attribute of an edge entity. The complete list of edge attributes are presented in Table 10. Curves of the type Line and Arc are discussed next.

Table 10 Edge Entities

Field	Data Type	Description
id	Integer	The edge id
surface_ids[]	Integer	Id's of surfaces adjoining this edge
directions[]	Integer	Edge direction for adjoining surfaces
uv_points[]	Double	The u and v values of adjoining surfaces
curve_type	Integer	Constant indicating the type of curve
curve	curve structure	The curve structure
uv_curves[]	structure	The curves on adjoining surfaces

Line

The line entity has the first and the last point coordinates as its parameters. The coordinates are of type double and have field names as *end1* and *end2*. The parametric equation of the line is:

$$P(x, y, z) = (1 - t) * end1 + t * end2$$

Table 11 Line Entities

end1[]	Double	First point on line
end2[]	Double	Last point on line

Arc

The attributes of the curve type arc have two unit vectors indicating its first and second plane vectors. Data is also stored for the arc's center point and radius. Other attributes include the start and the end angles of the arc (Table 12). The entity's parametric equation is given as:

$$c = (1 - t) * start\_angle + t * end\_angle$$

$$P(x, y, z) = radius * (cosine(c) * vector1 + sine(c) * vector2) + origin$$

Table 12 Arc Entities

Field	Data Type	Description
vector1[]	Double	Vector for the plane of the arc
vector2[]	Double	Vector for the plane of the arc
origin[]	Double	Origin of the arc
start_angle	Double	Start angle of the arc
end_angle	Double	End angle of the arc
radius	Double	Radius of the arc

BIBLIOGRAPHY

[1] Ahluwalia R.S., and Miller J.E., Dynamic Feature Generation During Design, *Journal of Design and Manufacturing*, vol. 19, 115-126, 1994.

[2] Binford T.O., Survey of Model-Based Image Analysis Systems, *The International Journal of Robotics Research*, vol. 1, 18-64, 1982.

[3] Chin R.T., and Dyer, C.R., Model-Based Recognition in Robot Vision, *ACM Computing Surveys*, vol. 18, 67-108, 1986.

[4] De Florianni L., Feature Extraction from Boundary Models of 3-Dimensional Objects, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 8, 299-314, August 1989.

[5] Di Manzo M., Trucoo E., Giunchiglia F., and Ricci F., FUR: Understanding Functional Reasoning, *International Journal of Intelligent Systems*, vol. 4, 431-457, 1989.

[6] Dixon J.R., Designing with Features: Building Manufacturing Knowledge into more Intelligent CAD Systems, *Proceedings of ASME Manufacturing International*, vol. 1, April 1988.

[7] Giacometti F., Chang T-C., Object-oriented Design for Modeling Parts, Assemblies and Tolerances, *Proceedings of Technology of Object-Oriented Languages and Systems (TOOLS)*, 243-255, 1990.

[8] Green K., Eggert D., Stark L., Bowyer K., Generic Recognition of Articulated Objects through Reasoning about Potential Function, *Computer Vision and Image Understanding*, vol. 62, no. 2, 177-193, September 1995.

[9] Henderson M.R., and Chuang S.H., Three Dimensional Shape Pattern Recognition using Vertex-Edge Graphs, *Computers in Industry*, vol. 22, no. 6, 121-126, July/August 1990.

[10] Joshi S., Chang T.C., Graph Based Heuristics for Recognition of Machined Features from a 3D Solid Model, *Computer Aided Design*, vol. 20, 58-66, March 1988.

[11] Kise K., Hattori H., Kitahashi T., and Fukunaga K., Representing and Recognizing Simple Hand-Tools Based on their Functions, *Asian Conference on Computer Vision*, 656-659, 1993.

[12] Pratt M.J., Synthesis of an Optimal Approach to Form Feature Modeling, *ASME Conference of Computers in Engineering*, 263-274, 1988.

[13] Qamhiyah A.Z., Venter R.D. and Benhabib B., Geometric Reasoning for the Extraction of Form Features, *Computer-Aided Design*, vol. 28, 887-903, 1996.

[14] Shah J.J., and Mantyla M., Parametric and Feature Based CAD/CAM, John Wiley, 1995.

[15] Stark L., Bowyer K.W., Function-Based Generic Recognition for Multiple Object Categories, *CVGIP: Image Understanding*, vol. 59, no. 1, 1-21, January 1994.

[16] Vaghul M., Dixon J.R., Zinmeister G.E., Simmons M.K., Expert Systems in a CAD Environment: Injection Molding Part Design as an Example, *ASME Conference of Computers in Engineering*, August 1985.

[17] Waco D.L., Kim Y.S., Geometric Reasoning for Machining Features using Convex Decomposition, *Computer-Aided Design*, vol. 26, no. 6, 477-489, June 1994.

[18] Werghi N., Fisher R., Robertson C., Ashbrook A., Object reconstruction by incorporating geometric constraints in reverse engineering, *Computer-Aided Design*, vol. 31, 363-399, 1999.

[19] Wilson P.R., Pratt M.J., A Taxonomy of Features for Solid Modeling, *Geometric Modeling for CAD Applications*, 1988.

[20] Woo T., and Tang K., Algorithmic Aspects of Alternating Sum of Volumes. Part 1: Data Structure and Difference Operation, *Computer-Aided Design*, vol. 23, no. 5, 357-366, June 1991.

ACKNOWLEDGEMENTS

The author would like to foremost thank Dr. Ranga Narayanaswami for his help and guidance during this research, without which this work would certainly have not been possible. The author also would like to thank other members of the program of study committee, Dr. Frank Peters and Dr. Soma Chaudhuri for their support and understanding in various matters related to this research. Lastly, the author would like to express gratitude and appreciation to family members and friends for their constant encouragement and motivation.